Air Force Institute of Technology
AFIT Scholar

Theses and Dissertations

9-14-2017

Evaluating Process Improvement Courses of Action Through Modeling and Simulation

Joseph R. Owens

Follow this and additional works at: https://scholar.afit.edu/etd

 Part of the Management Sciences and Quantitative Methods Commons, and the Systems Engineering Commons

Recommended Citation
Owens, Joseph R., "Evaluating Process Improvement Courses of Action Through Modeling and Simulation" (2017). *Theses and Dissertations*. 777.
https://scholar.afit.edu/etd/777

This Thesis is brought to you for free and open access by AFIT Scholar. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of AFIT Scholar. For more information, please contact richard.mansfield@afit.edu.

EVALUATING PROCESS IMPROVEMENT COURSES OF ACTION THROUGH MODELING AND SIMULATION

THESIS

Joseph R. Owens, Captain, USAF

AFIT-ENV-MS-17-S-049

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A.
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT-ENV-MS-17-S-049

EVALUATING PROCESS IMPROVEMENT COURSES OF ACTION THROUGH MODELING AND SIMULATION

THESIS

Presented to the Faculty

Department of Systems Engineering and Management

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Systems Engineering

Joseph R. Owens, BS

Captain, USAF

September 2017

DISTRIBUTION STATEMENT A.
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AFIT-ENV-MS-17-S-049

EVALUATING PROCESS IMPROVEMENT COURSES OF ACTION THROUGH MODELING AND SIMULATION

Joseph R. Owens, BS

Captain, USAF

Committee Membership:

Dr. J. M. Colombi, Ph.D.
Chair

Dr. B. T. Langhals, Ph.D.
Member

Lt Col A. M. Cox, Ph.D.
Member

AFIT-ENV-MS-17-S-049

Abstract

Quantifying an expected improvement when considering moderate-complexity changes to a process is time consuming and has potential to overlook stochastic effects. By modeling a process as a Numerical Design Structure Matrix (NDSM), simulating the proposed changes, and evaluating performance, quantification can be rapidly accomplished to understand stochastic effects. This thesis explores a method to evaluate complex process changes within Six Sigma DMAIC process improvement to identify the most desirable outcome amongst several improvement options. A tool to perform the modeling and evaluation is developed. This process evaluation tool is verified for functionality, then is demonstrated against generic processes, a case study, and a real world Continuous Process Improvement event. The application of modeling and simulation to improve and control a process is found to be a positive return on investment under moderate complexity or continuous improvement events. The process evaluation tool is demonstrated to be accurate in prediction, scalable in complexity and fidelity, and capable of simulating a wide variety or evaluation types. Experimentation identifies the importance of understanding the evaluation criteria prior to “Measurement” in DMAIC, which increases the consistency of process improvement efforts.

Table of Contents

Page

List of Figures

Page

List of Tables

Page

List of Acronyms

CPI	Continuous Process Improvement
COA	Course of Action
DMAIC	Define, Measure, Analyze, Improve, Control
DoD	Department of Defense
DOE	Design of Experiments
DSM	Design Structure Matrix
LSS	Lean Six Sigma
M&S	Modeling and Simulation
NDSM	Numerical Design Structure Matrix
ToC	Theory of Constraints

List of Definitions

Action (process)	One defined portion of work in a process to be performed after all inputs are met
Bottleneck	A term to describe the current task in a process that prohibits performance improvement more than any other task
Consistency	The expectation that evaluation will be used to choose the most beneficial course of action to the evaluation formula, and therefore efforts in earlier stages of process improvement should support evaluation
Combination	When analyzing a process that has multiple start or end tasks, and only one is used, a unique start and a unique end pair to form a combination
Confound	Measuring changes in a way such that differences can not be uniquely attributed to one factor
Constraint	A limitation preventing or slowing the execution of an action
Course of Action (process)	A suggested change to improve a process
Dependency	A directional relationships between two tasks
Evaluation	Determining the performance of a process
Evaluation criteria	The calculation to quantify the performance of a process
Goal (process improvement)	The desired outcome of a process improvement event
Improvement options	The set of potential changes to a process to attempt improved performance
Mandatory relationship	A graph or DSM dependency that must be followed to complete a process's objective

Mandatory task	A tast that must be accomplished to complete a process's objective
Objective	What a process was established to do or produce
Operation	One defined portion of work in a process to be performed after all inputs are met
Performance function	The mathematical representation of how a process combines resources, tasks, relationships, business models, and constraints
Process	"A series of actions or operations" (Webster, n.d.) that achieve a goal
Process Evaluation Tool	The set of code developed in this thesis to verify process improvement
Process improvement	The attempt to change a process with the intent of increase performance in one or more regards
Process improvement solutions	The different options proposed to change a process
Process manager	The individual owning or improving a process
Relationship data	Interaction data associated with each one-directional dependency inside the DSM
Resource	An object applied to a process to enable completion of tasks
Stochastic	Unpredictable statistically
Task	One defined portion of work in a process to be performed after all inputs are met task saturation

EVALUATING PROCESS IMPROVEMENT COURSES OF ACTION THROUGH MODELING AND SIMULATION

I. Introduction

General Issue

Often, an established process will need to improve its performance. When considering improvement, a process is often more than "a series of actions or operations conducing to an end" (Process, 2017); there are resources that perform actions, and at times limitations on the operations. Process improvement can be achieved by a great variety of options, such as increasing resources that perform actions, reducing the process's limitations, speeding up the actions in the process, automating portions of the process, reducing variability, or by editing the series of actions itself. Many approaches can be taken.

In 2009, the Department of Defense institutionalized the process improvement strategy of Continuous Process Improvement/Lean Six Sigma (CPI/LSS) (McGrath, 2009). Continuous Process Improvement is defined by Defense Acquisition University as "an integrated system of improvement that focuses on doing the right things, right" (ACQuipedia, 2017). CPI/LSS guides process improvement by eliminating tasks that do not add to the performance, identifying and controlling factors that cause variation in performance, or search for a "total performance solution" by applying Lean Six Sigma (Defense Acqusition University, 2016).

CPI/LSS encourages data-driven changes, using the Six Sigma method "Define, Measure, Analyze, Improve and Control" (DMAIC) (Nave, 2002). General Electric's Six Sigma implementation starts by defining the goals, problem, and boundaries, then creates and completes a plan to measure performance data. Analysis of the measured data identifies existing performance gaps and improvement opportunities. Improvement is then done by creating and implementing solutions to eliminate performance problems, and is finally controlled by preventing regression to the original state, promoting the new process, and closely monitoring performance. Through these steps, DMAIC seeks to establish a methodology to improve process performance without mandating a specific solution (DMAIC, 2017)

The step that implements a change—improve—is defined as "create [innovative] solutions using technology and discipline" (DMAIC, 2017). Measurement and analysis will have identified "what should be changed," but may not specifically identify how. If waste reduction and/or variance control will not improve the process, CPI/LSS calls for open-ended innovation.

During the Analysis and Improve steps of DMAIC, potential changes to a process are evaluated for expected process performance. Simple changes to a process can often replace one measured value with a goal value, and adjust the known performance by the change. In the case where "open-ended innovation" is required, evaluation becomes complicated.

"Innovation" implies that engineered solutions are less likely to replace a single task with an equivalent—but improved—task. When performance is being corrected across multiple tasks, the overall process requires full recalculation. Changes to a process

that are not specifically designed to target critical portions of a process are likely to cause a stochastic shift to performance.

The stochastic nature of process evaluation can be seen by viewing the portions of a process as components of the evaluation function. How the process's tasks are structured is a "plan of operations," which equates to the overall "shape" of the function. The process's tasks are either values or minor, independent calculations. The evaluation of the process combines the independent tasks into an equation in accordance with the plan.

For example, a six step process involving sequential and concurrent tasks is shown in Figures 1 through 3. To find the average time to complete the process, the sequential portions of the process are put in line (Figure 1), the rest of the process's structure sets the remainder of the equation (Figure 2), and finally the tasks themselves are put into the equation (Figure 3).

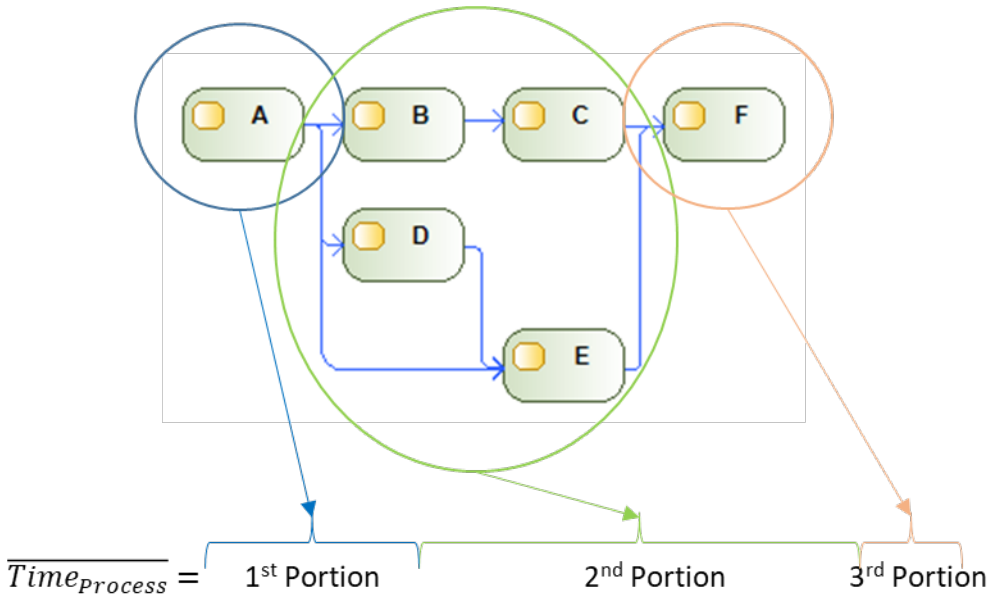


Figure 1: Visualizing Process Evaluation, Major Sequential Sections

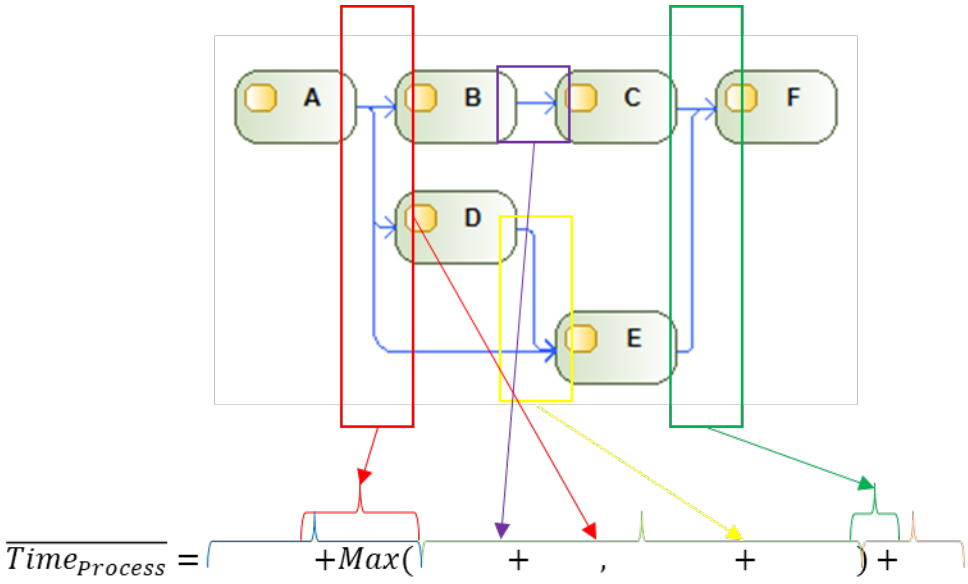


Figure 2: Visualizing Process Evaluation, Interdependencies

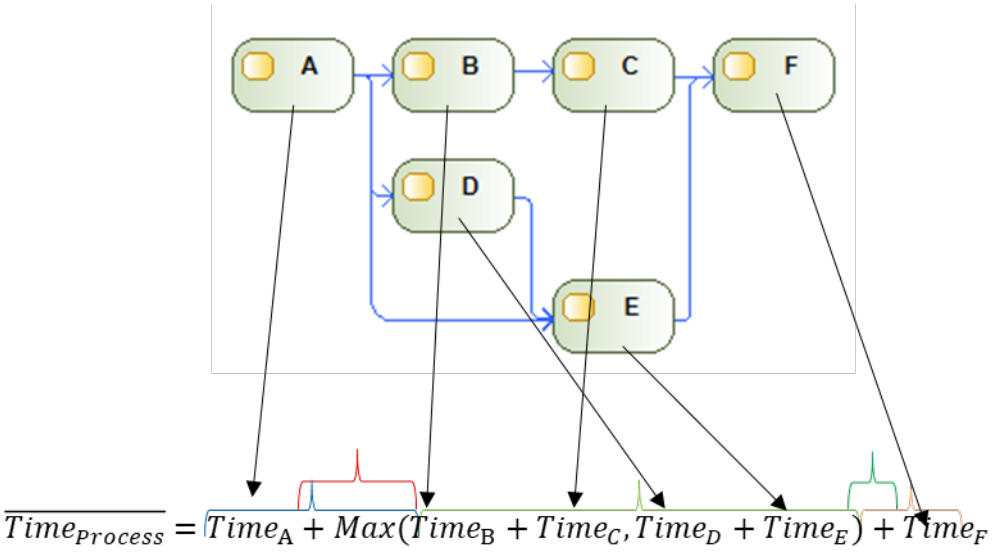


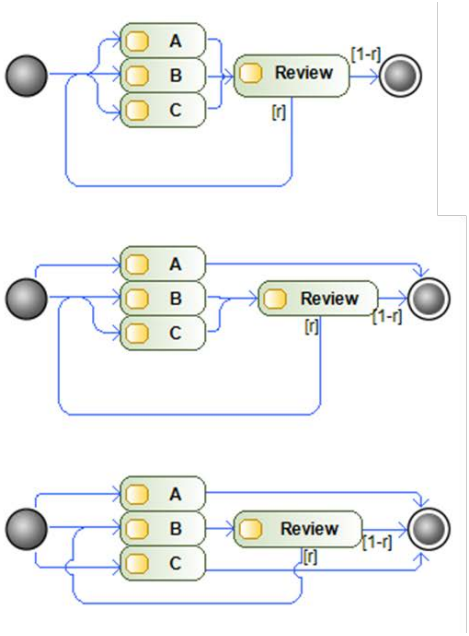
Figure 3: Visualizing Process Evaluation, Tasks

Changes to any of the tasks in the example would replace the value or calculation of the individual task. By contract, adding or removing tasks, changing several tasks at one time, or changing the connections of tasks Changes to the “plan of operations” would

change the existence and/or interdependent relationship of the calculations in the evaluation function (Biegler, 2010). Changing the structure of a process causes the performance to change in ways that are difficult to anticipate, and decide if the Course of Action (COA) should be used to improve a process.

An example of how a process's expected completion time can change without changing the actions follows. The process in Figure 4 has four tasks, tasks A, B, C and one task that reviews the work. The equation for average time to complete the process changes as the review's interaction with the three other tasks change. Holding the work in tasks A, B, or C constant, but varying what the review is allowed to impact, the time to complete changes.

Further, as Beigler stated, the equation to determine mean time changes as the process changes. When the process's structure changes, the objective function that defined the performance is rewritten. These rewrites do not guarantee a linear, extrapolated change to the objective function, and should be treated as a discrete and unique way of implementing the process (Biegler, 2010).



$$\overline{Time_{Process}} = \left(\frac{1}{1-r}\right)(Max(Time_A, Time_B, Time_C) + Time_{Review})$$

$$\overline{Time_{Process}} = Max\left(Time_A, \frac{Max(Time_B, Time_C) + Time_{Review}}{1-r}\right)$$

$$\overline{Time_{Process}} = Max\left(Time_A, \frac{Time_B + Time_{Review}}{1-r}, Time_C\right)$$

Figure 4: Expected Completion Time Comparisons

If the time to perform the work in A is 3 days, B is 2 days, C is 1 day, the Review completes in negligible time and is considered to be 0 days, and the Review has a 25% chance ("r") of causing rework, the mean time for the three examples is 4, 3, and 3 days respectively. Removing "A" from the review decreases average time by 1 day or 25%. Removing "C" from the review causes no change.

Without changing the work in any of the tasks, varying how the tasks interact results in performance shifts. The interactions of the tasks rewrite the function, and may or may not cause performance changes. When process improvement innovation will change more than a single task, it should be assumed that the objective function will change in a discrete manner.

Because DMAIC offers a methodology that may improve a process by innovation, changes are not easily evaluated for how much the process will improve. Without

formulating the process's evaluation function, the ability to decide how to improve a process is unclear. How does a process manager know their intended changes to a process will achieve the goals set when defining the process improvement? How does a process manager find better improvement solutions?

Problem Statement

In DMAIC, process improvement includes steps of measuring, performing analysis, and implementing improvements. The method to measure data and analyze it is not prescriptive, which enables LSS to improve a wide breadth of problems. However, the lack of specific guidance also allows innovative COAs to be selected without verifying that the COA is predicted to cause improvement. When complex modifications are made to a process, expected performance is less definitive. A method to accurately quantify complex improvements within DMAIC is needed.

Research Hypotheses

In DMAIC, simulation can be applied to evaluate if process improvements are likely to result in desired performance. Modeling a process's tasks and interactions captures the performance function, enabling simulation to replicate complex evaluation formula changes. Use of simulation in DMAIC should quantify the impacts of innovative courses of action, and allow selection of the best course of action presented during a process improvement event.

Investigative Questions

In order to improve process improvement when using the DMAIC method, simulation will be applied to quantify expected results. A reliable way to apply

simulation to a process is required to evaluate expected performance. In addition, a means to identify the best COA is desired. To be able to find the "best" COA, the simulation will need to be able to evaluate a wide range of improvement goals types and quantify each. To update DMAIC, two primary questions will need to be answered, and best practices identified during research should be documented.

1. What is the formulation to evaluate process improvement courses of action?
2. How does evaluation find the best way to improve a process?
3. What heuristics from evaluation guide process improvement?

Methodology

Modeling and simulation will be used to evaluate processes. The generic improvement goal will be to minimize the mean time to fully perform a process while reducing variance. The process models will include tasks, sequencing, resources, and constraints. Analysis will be performed on both the modeled behavior in addition to standard DMAIC analysis. Comparisons of the original process and COAs will be used to determine what improvements exist, and what the expected benefit should be.

Assumptions/Limitations

Due to the breadth of ways that process improvement can be implemented, assumptions and limitations must be applied to ensure that COAs' evaluation is comparable.

- Improvements that completely redesign a process are outside the limits of this research. It is expected that measured data will not be applicable to a top-down

reengineering of a process, and simulation would not be able to predict future performance.

- It is assumed that the relationships between a process's tasks are valid and necessary, unless it can be reasonably deduced that the relationship was incorrectly documented.
- It is assumed that the process's goal will have been completed if all mandatory tasks are successfully completed at least one time after their prerequisite mandatory relationships have been performed, and no work remains to be done.
- It is assumed that to complete a process, improvements cannot remove mandatory tasks.
- It is assumed that process performance data omitted from the process model is not necessary for evaluation at the required fidelity (ie a learning curve that was not modeled because it would not significantly impact evaluation results).

Expected Contributions

It is expected that this method will allow moderately complex COAs to be quantitatively evaluated. By applying the method to several proposed solutions, quantitative comparison should facilitate better decisions. By exploring modeling and simulation in DMAIC, a method to evaluate process improvement COAs and identify the best option will be developed, verified, and demonstrated.

Preview

This thesis is organized as follows. Chapter II summarizes established process improvement techniques used in the Department of Defense (DoD), limitations in process

optimization, evaluation methods, and industry applications of modeling and simulation for processes. The research topics from Chapter II are applied in Chapter III to develop a process evaluation tool, the tool is verified and validated, and the experimentation methodology is established. The four stages of experimentation with the process evaluation tool are performed. The process evaluation tool is used to find improvements in notional processes, recreate a documented process improvement event to see if measured outcomes could have been predicted, and applied to a real-world process improvement event to correct issues. The problems experienced with the process evaluation tool are also summarized. In Chapter V, the research questions are answered, the value of modeling and simulation in process improvement is summarized, and recommendations for further simulation refinement are made.

II. Literature Review

Chapter Overview

This chapter researches the topics that are necessary to combine process improvement techniques into a modeling and simulation tool. Current guidance and techniques are briefly covered, current limitations of simulation are given, and a history of research and implementations of process simulation is covered.

Process Improvement Methods

With a nearly limitless variety of process issues that need to be solved, many techniques exist to achieve improvement. The military currently instructs the use of Continuous Process Improvement/Lean Six Sigma (CPI/LSS), to include Business Process Reengineering (BPR) (McGrath, 2009). Within CPI/LSS and BPR, there are four primary tenants to improve a process; CPI, Lean, Six Sigma, and BPR

The primary method used in this thesis is Six Sigma. Six Sigma focuses on "minimizing waste by reducing and controlling variation" within a process (ACQuipedia, 2017). Six Sigma is often performed through the five step process "DMAIC" previously described. Six Sigma is data driven, measuring task and process performance multiple times to determine variance. This methodology is incredibly powerful in manufacturing and queueing processes that are continuously run, but also can apply to nearly any process that is performed more than one time.

Additionally relevant to this thesis, Continuous Process Improvement (CPI) can be applied to update and Control a process as the recursive last step of DMAIC. CPI is establishing groundwork to recursively assess and improve processes through

incremental improvements over time (ACQuipedia, 2017). In modeling and simulation, an established process model can be used to evaluate if an emerging COA is beneficial. When CPI evaluation of new COAs is compared to previous options in an identical model, quantitative expectations are given a frame of reference.

While CPI is vague by ACQuipedia's definition, the Cosima Project's definition of the synonymous "Kaizen" is clearly stated. Kaizen is "a management concept, which focuses on the gradual improvement of processes and on the development of people so that they are able to solve [problems]. It's not a project, it is a comprehensive tool and mindset to develop the business" (COSIMA, 2015, p. 6). CPI does not itself improve a process, but sets a plan to generate and integrate multiple small improvements over a long period of time.

Two concepts within CPI/LSS—Lean and Business Process Reengineering—are not applicable to simulation to evaluate COAs. They are described below, along with an explanation as to why they are excluded.

The first excluded CPI/LSS principal is Lean. Lean analyzes a process's workflow, identifies what is "value" to the process objective and what is not, then remove the non-value work. Conceptually, Lean can be applied to both manufacturing and processing as well as management philosophy (COSIMA, 2015, p. 5).

When developing a model for computerized analysis of a system, an assessment of a task's "value" would need to be established prior to simulation for a computer to evaluate value, and would be redundant. Lean principals should be applied prior to establishing a model of the system.

The other method excluded from this thesis is BPR. BPR sets out to completely redesign the process from the top down, by starting with the objective of the process. In military definition, BPR "starts with a high-level assessment of the organization's mission, strategic goals, and customer needs. […] Only after the organization rethinks what it should be doing, does it go on to decide how best to do it." (Dodaro & Crowley, 1997, pp. 5-6).

The scale of BPR is large enough that applying known performance to a completely redesigned process would be problematic. To be able to make comparable evaluations via simulation, similar accuracy between each COA would be required. BPR should be performed to achieve a new paradigm in the process, and previously measured data should not be expected to be comparable to future performance after BPR. As such, large changes to processes have been specifically identified as a limitation in this paper's Assumptions and Limitations.

Optimization of Processes

While this thesis focuses on the evaluation of COAs already theorized on generating process improvement, understanding improvement as it relates to process will assist in answering the second research question. Optimization of a process is finding the process changes that will give the evaluation as close to the desired value as possible while still achieving the objective of a process. In most cases, the solution would be to maximize or minimize the output of an equation. For processes, the equation to be optimized is the evaluation criteria.

Two issues complicate optimization of a process. First, the objective formula to calculate process performance may be poorly behaved. The second is that process improvement has the option of rewriting the objective function.

Consider a process as a series of tasks that rely on one another to achieve a goal. In Theory of Constraints (ToC), one of those tasks will be the bottleneck that limits performance more than the others. By improving other tasks, the process may improve, but only so far until the current bottleneck is eliminated. When the task is improved enough that it is no longer the bottleneck, the bottleneck for the process shifts to another task (Vorne, 2016).

Under ToC, improving a task by one minute can result in up to one minute of improved performance. Applying an improvement to a task that is not the bottleneck may result in little or no improvement. Applying improvement to the bottleneck will favorably impact the evaluation, but once the bottleneck shifts the process may stop improving.

Depending on where and how improvements are made to a process, the input to process improvement will have different performance impacts. The bottleneck shifting causes the relationship between change and improvement to be a non-continuous function. Because the gradient of the function is unreliable, ToC makes process improvement poorly suited to optimization outside small regions.

The second issue is that evaluation function changes as the process is rewritten. As demonstrated in Figures 1 to 3, changes to the process's structure cannot be calculated under the same function. When changes are made to the process structure (adding, removing, or altering relationships of tasks), there are discrete changes to the evaluation (Biegler, 2010).

Changing the structure of a process results in an evaluation equation that needs to be treated as a unique entity. In this case, optimization requires one of two choices. Either a local optimum needs to be accepted, or all discrete possibilities must be evaluated (Biegler, 2010).

Due to the limitations on optimization, the evaluation tool developed for this thesis will not attempt to optimize a process. Instead, a set of potential ways to improve a process will be given to the tool, and evaluation of every COA will be performed to find the best option given.

Methods to Evaluate Process Performance

To quantify process performance, a measurement criteria is needed. Criteria for measurement is determined by the improvement goals, whether the evaluation is on efficiency, cost, time, etc. Furthermore, an improvement goal must be set to be able to compare one process with another. In most cases, comparison will be looking for either a minimum or a maximum performance returned by evaluation to enact a decision aimed at improving a process.

Some well-behaved process evaluation functions can be solved mathematically. If the system can be broken down into a mathematical representation, it's possible to solve for system eigenvalues and find a true optima (Yassine, Jolekar, Braha, Eppinger, & Whitney, 2002). The process's task and relationship structure lends itself to a second option; solving a Markov Chain of equations to uniquely define the variables impacting the performance of a system (Carrascosa, Eppinger, & Whitney, 1998, pp. 1-10). For

evaluation functions that do not change, Genetic Algorithms and Synthetic Annealing can determine optimal performance.

In an environment where the process's structure changes, the evaluation function is rewritten. To know how a process will perform after complex changes, the process would need to be re-converted into a function, and the function reevaluated.

Another option would be to bypass maintaining the process's evaluation function as a whole, and simulate the completion of the process. Performed a significant number of times, Enrico Fermi's Monte Carlo simulations can approximate the function's output. In spite of complex interactions, the use of computer automation lends itself to simulating a dynamic system model. Using a simulator to replicate the impact of relationships, process performance can be evaluated.

The Design Structure Matrix

To leverage computer evaluation of a process, a digital model of process information is needed. Processes have complex decisions, reviews, and iterations which need to be methodically organized, or more simply stated, converted into structure for calculation. The fundamental structure of the process are the actions and their relationships (Larman, 2005) as seen in a diagraph, so a digital corollary of a diagraph is ideal.

Table 1: DSM Terminology Correlation (Different DSM Types, 2016)

Graph / Flow Diagram Terminology	DSM (Generic) Terminology	DSM (Task Based) Terminology
Action Entity Node Operation	Element	Task
Relation Edge	Dependency	Dependency Relation

The required conversion can be achieved by establishing a matrix of N process tasks (generically called “elements” in a DSM to refer to a row or column) on both axes of a matrix in an identical order and logging the interactions between tasks as “dependencies” between elements at their intersections in the matrix, (Figure 5). The completed matrix is a 2 dimensional structure containing tasks and interactions as elements and dependencies (respectively) correlating to a process flow diagram’s steps and linkages, respectively. This matrix is referred to as a Design Structure Matrix, or DSM.

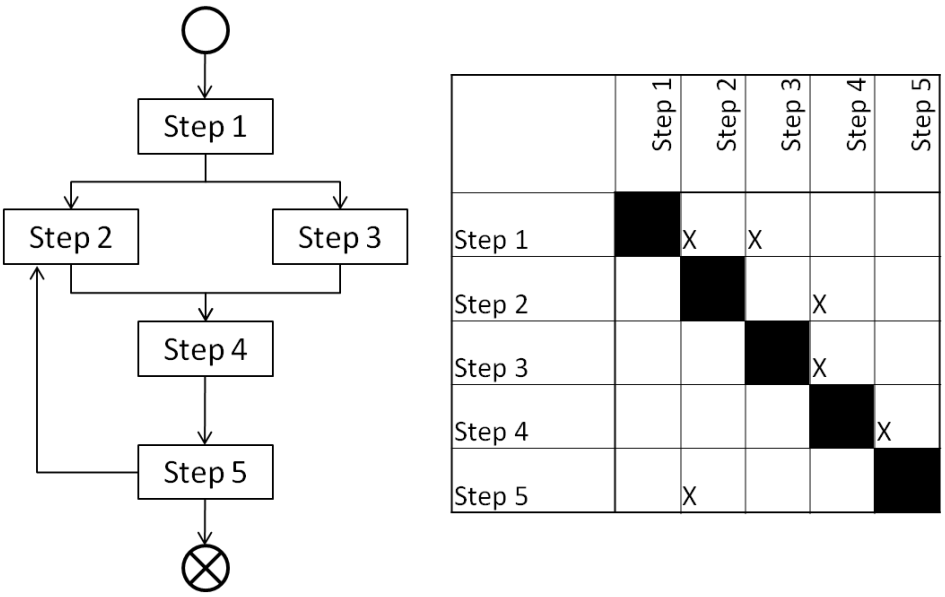


Figure 5: Process Flow Diagram to DSM Representation

In Figure 5, a five step process is converted from a flow diagram (left) into a DSM (right). The steps that are entered into the DSM are identical between the X and Y axis. This ordering ensures that each potential link between steps is available twice, meaning that directionality can be tracked on the relationships (ie Step 1 leading to Step 2 is in a different location in the matrix than Step 2 leading to Step 1). The diagonal on this DSM is blackened out to indicate that steps are not able to depend on their own actions. If a process can have tasks that can iterate autonomously, the DSM's diagonal can be used to capture the linkages.

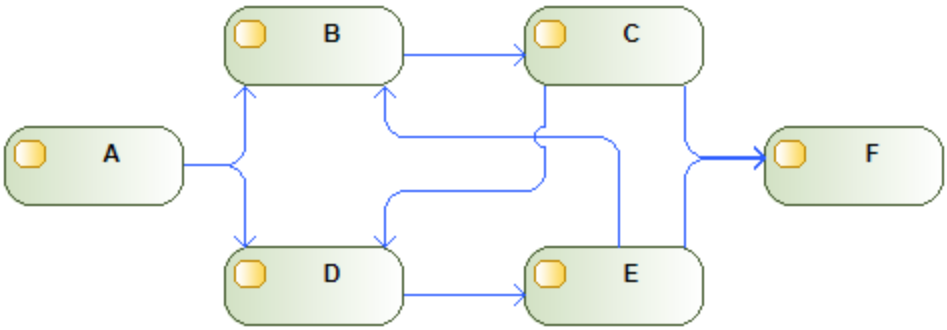


Figure 6: 3-Dimensional Feedback Loop Example

In the example from Figure 6, there is no linear process that can accomplish the process due to indirect codependent tasks. Based on different execution plans for the process, several implementations could be set up with the understanding that reiteration is inherent to the process. The method of when and where to accept the reiteration is handled in the implementation (Figure 7). Below are three nodal drawings of the process being implemented and executed three different ways: linear with a single feedback, concurrent with two feedback loops, and concurrent with a single feedback loop. To the right of each nodal drawing, the DSM for the process as drawn is shown.

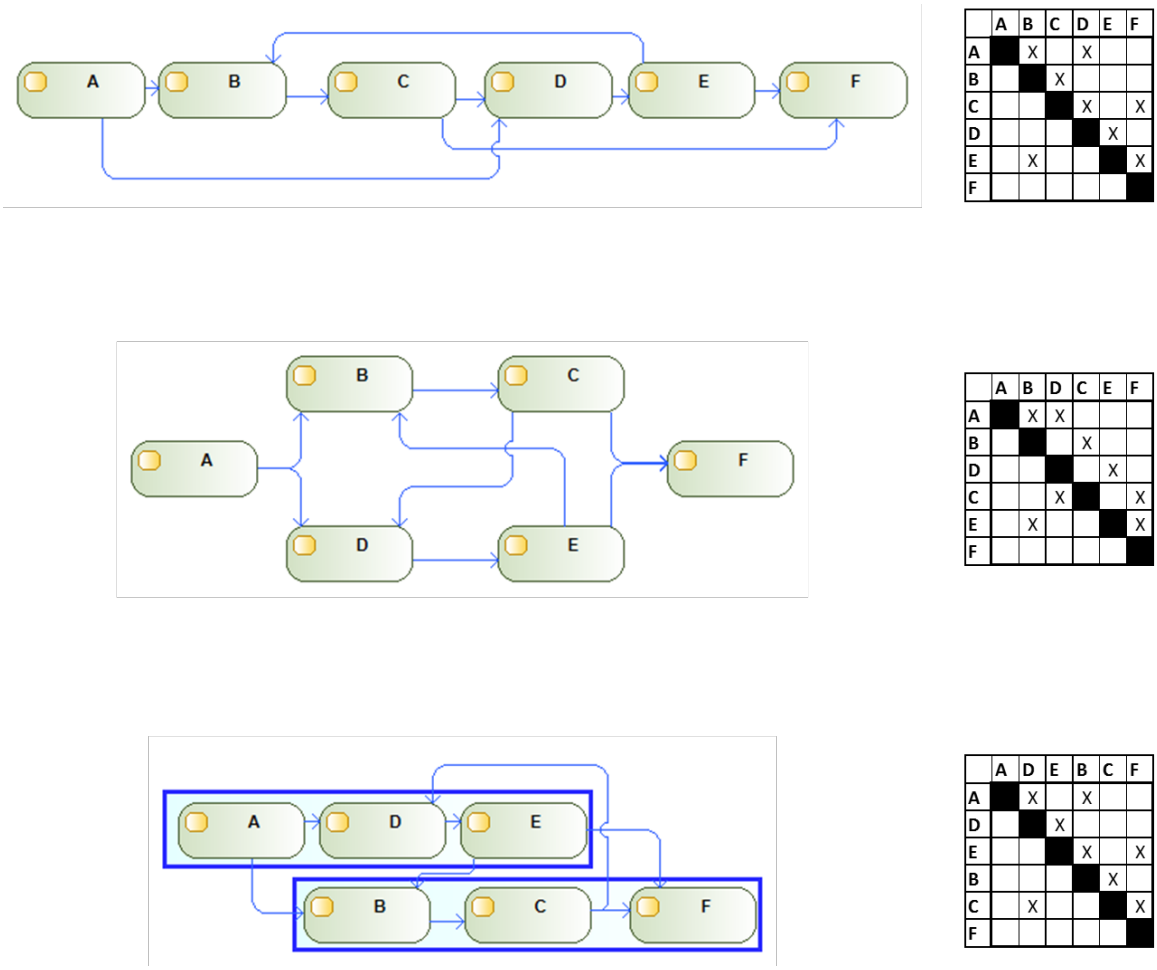


Figure 7: Three Implementations of One Process with Corresponding DSMs

The DSM's critical feature is that a correctly documented matrix will maintain the same data of tasks and relationships regardless of the ordering interpreted from a digraph. In Figure 7, the task ordering changed to reflect the implementation of the process, but the relationships between each element remain persistent. The inverse is also true; reordering the DSM's elements while maintaining the relationships will still accurately depict the digraph.

The ability to manipulate the DSM's axis order allows free alteration the task execution of the process while preserving relationships between tasks. With a DSM

representation of a process, a single model can represent every possible implementation of the tasks and relationships.

As the DSM is reordered, if the elements are executed in order from top to bottom, it represents an ordered implementation of the process's tasks. Within feasible DSM orderings, different evaluation results may be found for the same process. It is possible to order the tasks in a way that does not follow relationships, or does not achieve a goal per the definition of a process, so validity of a DSM's implementation order must be checked.

Numerical Design Structure Matrix

DSMs were used to document the existence of interactions between tasks, but additional functionality of DSM was gained by including quantitative information. In Yassine and Falkenburg's mathematical analysis of tolerance control, a DSM tracked the relationships between subsystems, and was combined with measured data and a separate model to calculate the impact of subsystem manufacturing tolerance to the final product reliability (Yassine & Falkenburg, 1999, pp. 223-234).

Steward suggested using a weighting scheme internal to a DSM's relationships showing the order of how important the relationship was. Early alterations were to add the "strength" of the relationship, typically by using a number. The resulting numerical design structure matrix (NDSM) contained more information, and could help decision making (Steward, 1981, pp. 71-74). For example, McCord replaced subsystem relationships with weights, and applied sorting algorithms to minimize "distance" from the diagonal overall to the DSM. The resulting DSM showed several subsystems that

would need to frequently interact, indicating that the business's integrated team should be restructured (McCord, 1993, pp. 22-38).

Rather than limit data to one number for each relationship, additional data began to be imbedded using sub-matrices (Figure 8). Two issues arose from this implementation. First, the complexity of the data structure become cumbersome, and secondly data associated to the *element* rather than the *relationship* was being redundantly added to every cell (Figure 8). In this figure, four types of data are in each relationship; a probability to follow a specific relationship (%), the number of days of manpower required to perform the task (d), the number of people that can simultaneously work on the task (p), and the derived time to perform the work on a task (t = d/p). A large matrix with several types of data per relationship becomes difficult to analyze.

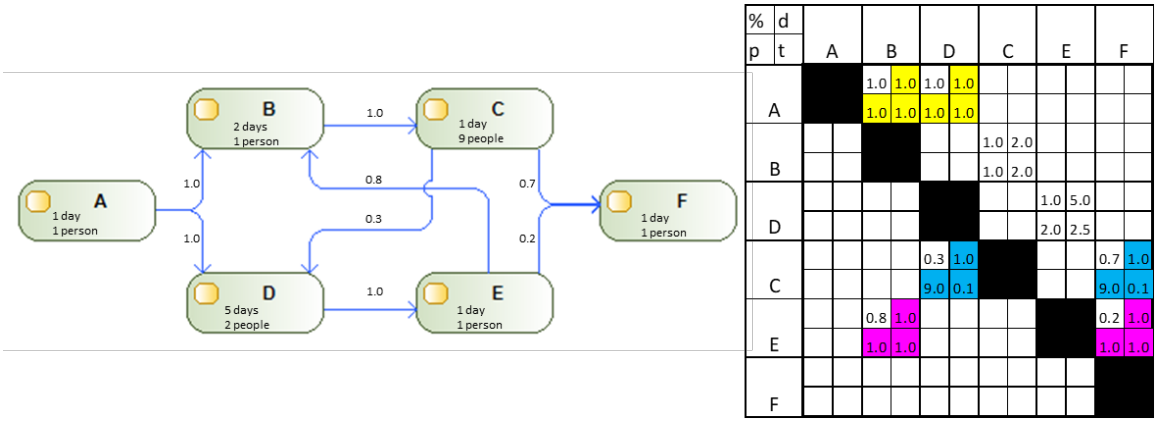


Figure 8: Example NDSM with Redundant Data Highlighted

Redundant data was eliminated by extending one axis beyond the element model, and adding data beyond the elements. In Figure 9, days, personnel, and time are pulled outside the dependency matrix of the NDSM as they pertain to the task, while relationship data remains internal to the matrix. Additionally, performance data for task F

was not available in Figure 8, as F did not have any outgoing relationships. By placing element data on the row and relationship data internal to the NDSM, the data more accurately represents the process.

%	A	B	D	C	E	F	d	p	t
A		1	1				1	1	1
B				1			2	1	2
D					1		5	2	2.5
C			0.3			0.7	1	9	0.1
E		0.8				0.2	1	1	1
F							1	1	1

Figure 9: Extending Task Data beyond N by N DSM

Process Simulation using NDSMs

The NDSM established itself as a powerful analysis tool to evaluate and improve engineering models. Several studies in business and education began using DSMs for simulation-based analysis, and developed tools from DSM simulation. Four examples of these follow.

Yassine, Whitney, Lavine, and Zambito experimented with modifying DSMs by applying heuristics and evaluating the functionality of the DSM itself. The analysis translated a design process into a model that simulates cycle time. Manual manipulation of the DSM indicated highly coupled elements in the process. Changes to these elements split into more manageable, less coupled steps with large reductions (28%) to the ways that a process could iterate. Additionally, they reduced the “length” of each iteration, so that if rework were required it would not require as much effort. The study identified a key tenant; to improve process time, reduce error probability and/or shorten unavoidable

integration loops by starting with near-correct work up front (Yassine, Whitney, Lavine, & Zambito, 2001, pp. 1-8).

Browning and Eppinger's analysis on project management via DSM models showed that combining rework likelihood against duration and cost of tasks allowed project tools to control budgeting, deadlines, and baseline a project. They also identified process improvements to be applied to the system analyzing cost versus schedule tradeoffs (Browning & Eppinger, 2002, pp. 428-439).

A method for managing a schedule by correlating tasks, relationships, and resources was accomplished in a system called DEPLAN. Taking a process model and evaluating CPM/PERT analysis with an activity based DSM provided a Gantt schedule for steps within a process (Choo, Hammond, Tommelein, Ballard, & Auston, 2004, pp. 313-326).

In spite of the major successes in the late 1990s to present date, there have been several limiting factors on the capability of simulation using NDSMs to facilitate analysis. Calculation of the system of equations is not always possible due to processing limitations on large matrixes. With reduced fidelity due to processing limitations, simulation/optimization end time may not be the optimal; algorithms may not select the lowest value calculated or possible (Thebeau, 2001, pp. 27-55). These issues will be handled by using Monte Carlo simulation rather than calculation, and evaluation rather than optimization.

Summary

Military use of CPI/LSS allows for various methods to achieve process improvement. However, the specific implementation for improvement may cause unforeseen consequences as bottlenecks and relationships shift. Anticipating changes to a process's performance are difficult to model, but the resulting process is still relatively easy to model. An NDSM can establish a digital model of a process independent of implementation, allowing evaluation of a process and modification of the process model. Analysis of the process model, simulated evaluation of proposed changes, and analysis of the changes can identify the best option, if every option is evaluated.

III. Methodology

Chapter Overview

This chapter covers the development and experimentation in augmenting process improvement with modeling and simulation. It starts with the steps to be taken to answer research questions posed in the first chapter. Chapter three focuses on development of a tool to evaluate a proposed process change—or Course of Action (COA)—to quantify performance, and verification of the tools' functionality. The chapter concludes by elaborating on experimentation procedures and notes.

Research Methodology

To be able to improve a process's performance, a basis for evaluating performance and a measure of "goodness" will need to be developed. To evaluate a process, code that simulates a process model and returns the performance data will be used. Simulating the baseline process and a series of changes to the will determine which option is the best. The combination of the model, simulator, and comparator will comprise the process evaluation tool.

To understand how to apply modeling and simulation (M&S) to DMAIC, four stages of experimentation will be used. First, existing processes will be simulated to search for types of improvements the tool can discover. Secondly, process improvement case studies will be used to understand the accuracy of the evaluation, and when modeling and simulation will fail. Case studies where the tool failed to correctly predict an improved process's performance will be researched to define limitations of modeling and simulation in CPI/LSS. Finally, using the tool against real-world military process

improvement events will help define how DMAIC needed to be modified to incorporate modeling and simulation.

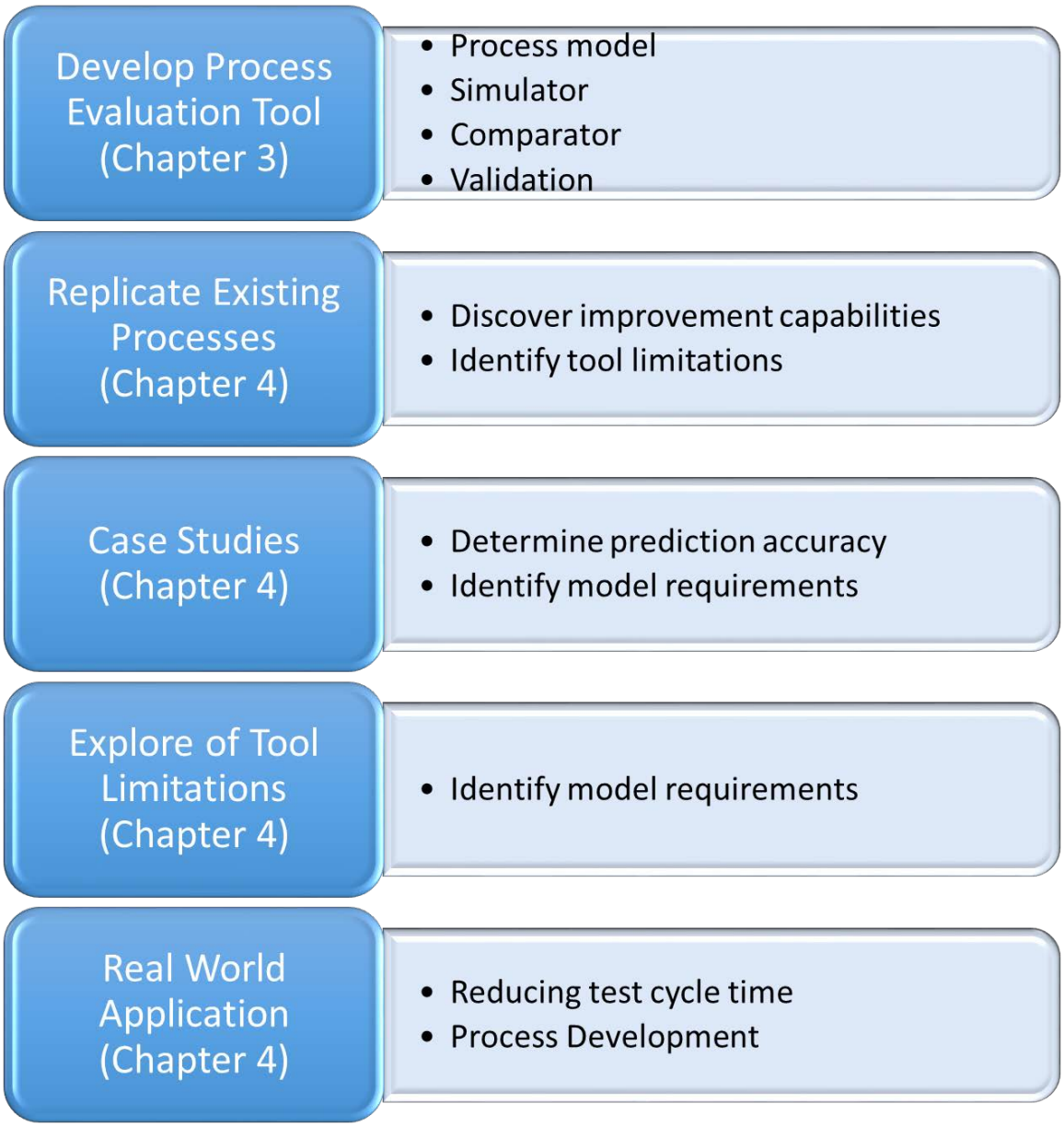


Figure 10: Research Methodology

Process Evaluation Tool

To identify the best improvement option for a process, three components are required: a model of the process, a simulator to run an instance of the model, and an evaluator to compare different COAs. This section outlines the logic behind the architecture in these three sections, and describes how they work.

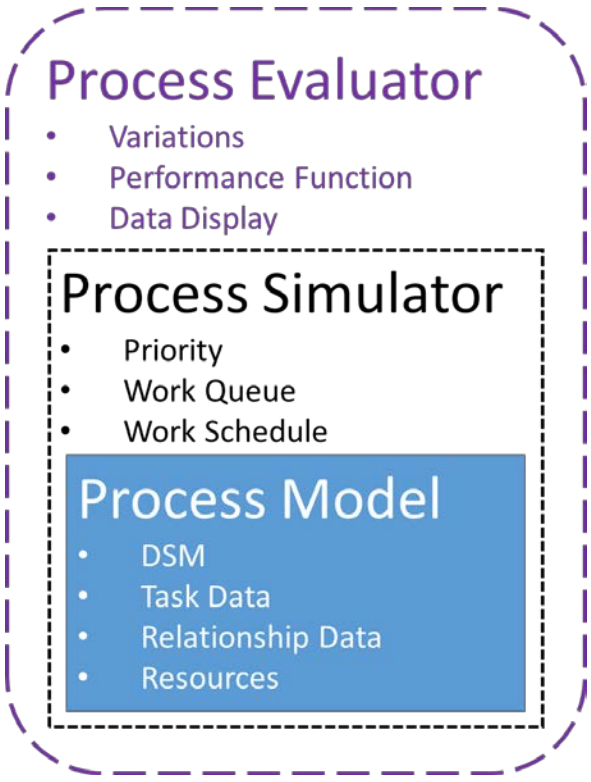


Figure 11: Process Evaluation Tool Architecture

Process Model

The model represents the process that needs to undergo improvement. The model is divided into four portions: the Design Structure Matrix (DSM) capturing tasks and dependencies, task data, relationship data, and resources that perform the tasks. The process's data is implemented in Matlab as a series of purely numerical matrices.

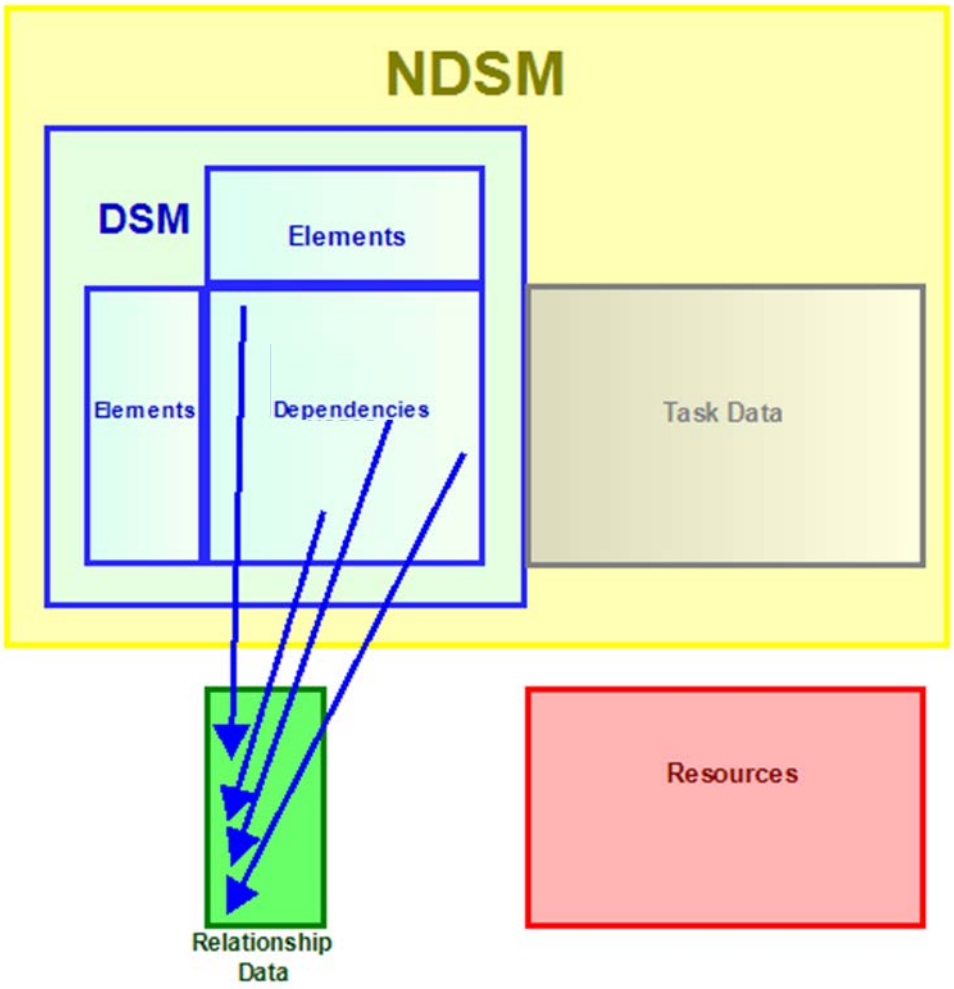


Figure 12: Process Model Architecture

Applying lessons learned from II. Literature Review, a Numerical DSM (NDSM) will be used to capture the location of relationships and task data. The DSM captures a process by structuring tasks as elements and task interactions as relationships. Task data is stored to the right of the DSM by extending the X axis. Critical information stored in the task data includes the original DSM order, how many hours a task takes on average, a value—if used—for randomization on the task time, what resource/s are used on a task,

and the maximum number of resources that can simultaneously work on one instance of the task.

In addition to the task data, the data on how tasks relate to one another must be modeled. Relationship data is the interaction data associated with each one-directional relationship inside the DSM, rather than the data that is associated with a task or the data indicating the existence of a relationship inside the DSM.

Relationship data was added separately from the DSM's N by N matrix to account for the nuances of simulating certain types of tasks and relationships. For example, a graph can insert a decision that chooses one of several outcomes as a logical element between tasks. The choice to add relationship data was to avoid generating several tasks in the DSM to capture the data only associated with the one chosen outcome of a decision point. Associating relationship data with tasks also had issues when the process would split as a result of one task queueing multiple tasks.

With simulating a process DSM, a decision would either need to be an independent task with no attributes except queueing one of multiple outcomes, or would need to be stochastic data associated with the relationships exiting a task. The latter option was chosen, and was implemented as a separate matrix of data referenced by the relationships inside the DSM.

The data relevant to how the tasks are accomplished, but common across the process is stored external to the tasks, in a resources matrix. The resources matrix was added to account for resources that perform work shared across multiple tasks within the DSM. Examples of necessary resources include personnel, personnel with specific skill

sets, and specialized tools that are shared in multiple tasks such as a backhoe in a construction project.

Finally, resources common to all tasks were stored in a separate matrix. Each resource had its own column, and for each unit of time, the availability of the resource was tracked. This was done to ensure that—as the process would task resources in priority order—no resource was working on more than what it could handle. A good example is manpower; if all hands are working the top priority task with all their effort, no other tasks should be performed.

Process Simulator

The simulator receives a process's model and a task priority, evaluates the process, and returns performance data. Evaluation is performed by implementing a priority work queue and a schedule, then simulates the execution of all tasks in a resource constrained environment.

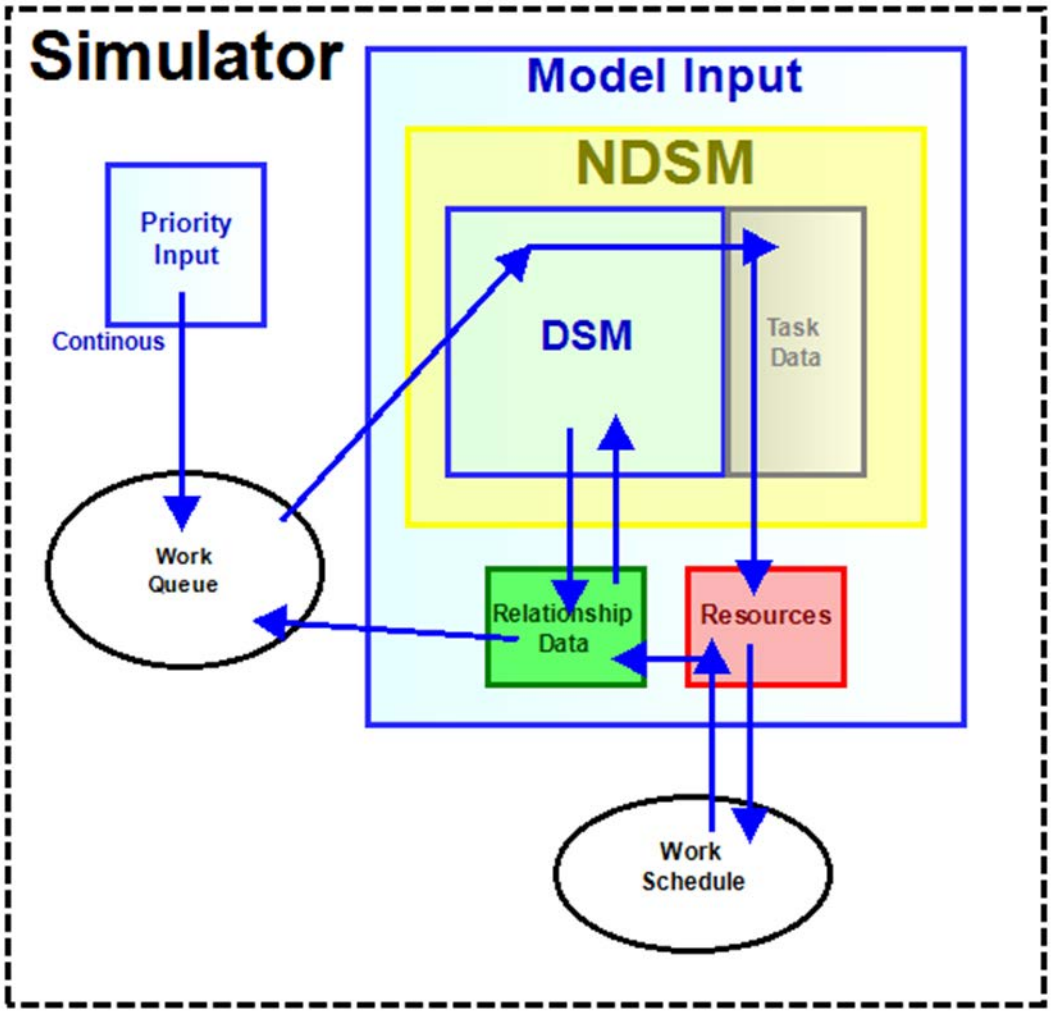


Figure 13: Simulator Architecture Diagram

At a top level, process simulation is achieved by taking the process model, queuing tasks to be worked upon through the relationships, tasking resources to complete the task, checking the relationships to queue additional work (if the model would queue work), prioritizing the work, and iterating until all work is performed. Once the process has completed, the history of the work queue and resource schedule provides metrics on process performance. The historical data is exported for use in the evaluator.

Work enters the queue when an element completes its task, then determines which relationships (if the relationships would execute stochastically) are activated in the simulation, and what elements will need to be queued for work as a result of the relationship activated. If no work is in progress or queued, the simulator will queue the first incomplete element. The simulator will not begin work on an element unless there is enough qualified manpower to complete the task, and predecessor tasks (ie mandatory relationships leading into the element) have been completed.

Once all mandatory elements have been performed at least one time after satisfying necessary predecessors, and no work is in queue, the process is assumed to be complete. Upon completion, measurements of the tasks performed and application of resources can be taken from the work queue and the resource schedule. Multiple Monte Carlo simulation runs are used to approximate statistical performance for the process.

It should be noted that the simulator follows the rules in the constraints and assumptions; all tasks are necessary, included in the model, must be completed, and have correctly documented relationships. As a result of these assumptions, orders passed to the simulator that do not pass feasibility criteria will execute the process as close to the desired order as possible. This will be a constraint in the optimizer's success later in this chapter. It is possible to simulate a successful process execution that does not meet all feasibility criteria, but requires manual review and cannot be compared in an optimizer.

Evaluator

The evaluator takes a statistical sample of simulation data, extracts the data to evaluate the desired performance function, and provides a quantitative value for the performance of the process model that was simulated. The evaluator accepts changes in

portions of the simulation are treated as different COAs, each COA is evaluated separately, and results are displayed as side-by-side box plots and mean value.

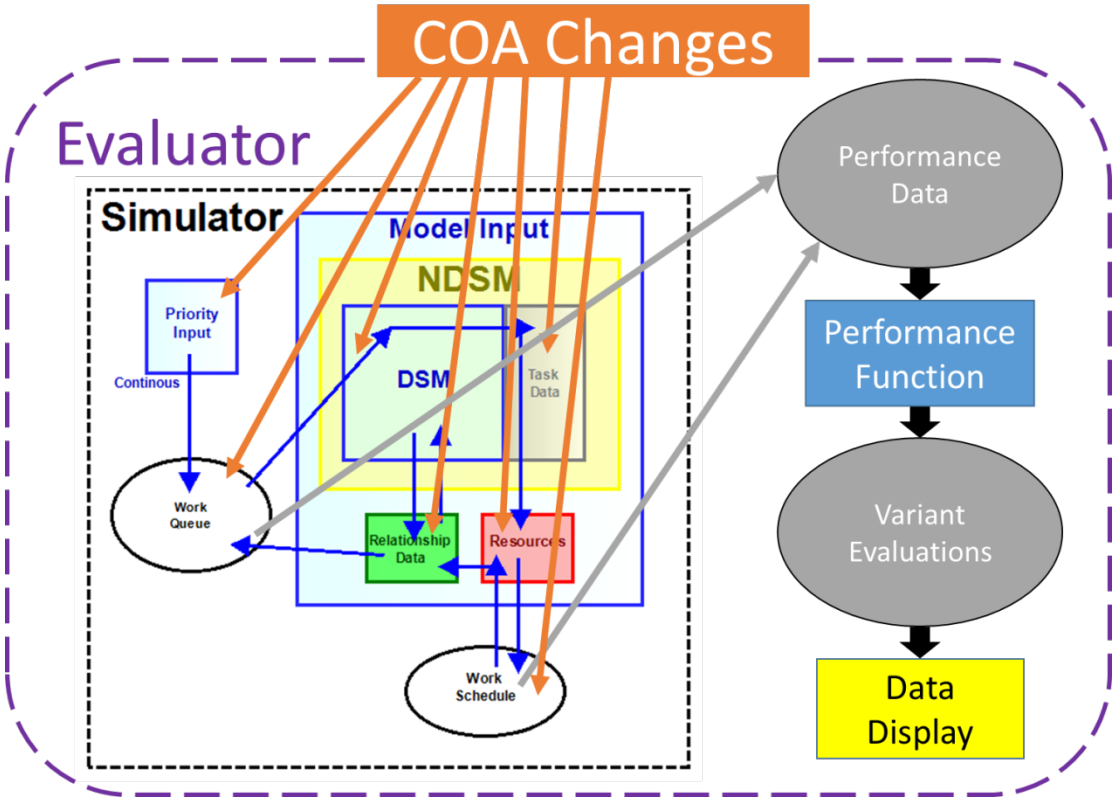


Figure 14: Evaluator Architecture

Evaluation takes one or more combinations of unique NDSM, task priority, resource data, and relationship data as an input. Each of these can be changed to evaluate performance for a potential solution. Multiple changes provides variation to the evaluator, and comparing evaluation of each COA helps identify a specific set of process, resources, and prioritization that is the local optima.

Each COA is simulated several times, and each individual simulation contributes to a statistical understanding of process performance. This is the same as William Gosset's work in sampling the Student-T distribution, where no assumptions are made

about the distribution's shape. For each COA, the evaluator extracts the data from process simulation, applies the performance evaluation, correlates the evaluation to the COA, and displays the performance for each combination side by side.

Verification of the Modeling and Simulation Code

To verify the functionality of the code, a buildup procedure was followed. The end goal is code that handles both deterministic and stochastic elements of both tasks and relationships accurately in a prioritized and resource-constrained environment for several COAs.

Validation began with simulating a simple, deterministic problem with a known value. The verification then added a range of completion time, and finally feedback loops. Then, deterministic, probabilistic, and iterative data will be combined to show that evaluation is returning expected results. With each addition of complexity, the process is simulated 10,000 times to show the distribution curves follow expected values. Verification started using a simple 3-step process Figure 15, having each step take 1 day exactly. Simulation shows that the process will always take 1 day per step, and have no stochastic performance (Figure 16).

Figure 15: Deterministic Process for Verification

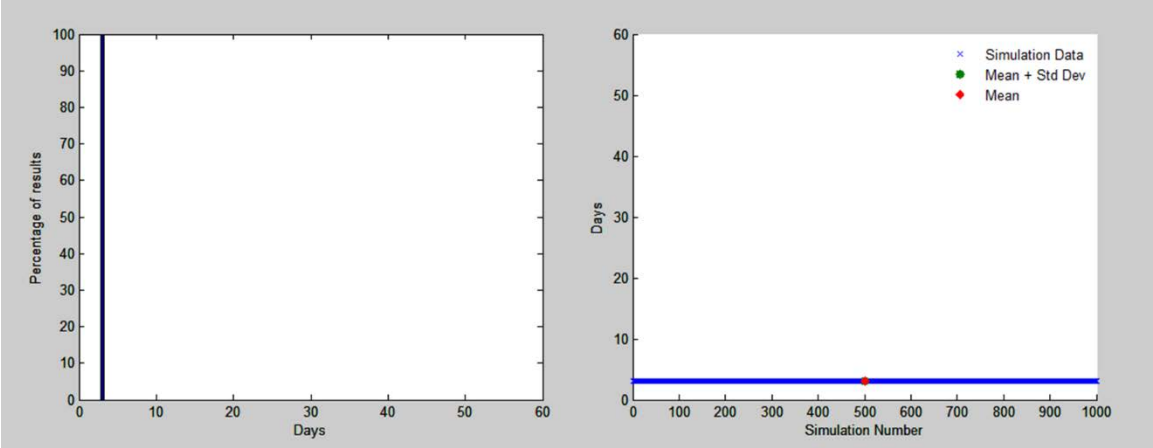


Figure 16: End Date Histogram (Left) and Scatter (Right) Plots for Figure 15

When using one day as the smallest value, any amount of work will round up to the next day before queueing the next task. By adding a random value between -0.5 and + 0.5 days to each task, each task should take one or two days, and an expected result is a range from four to seven days. Simulation roughly produces a bell curve from four to seven, with mean of 5.5 days.

Figure 17: Process With Task Distributions

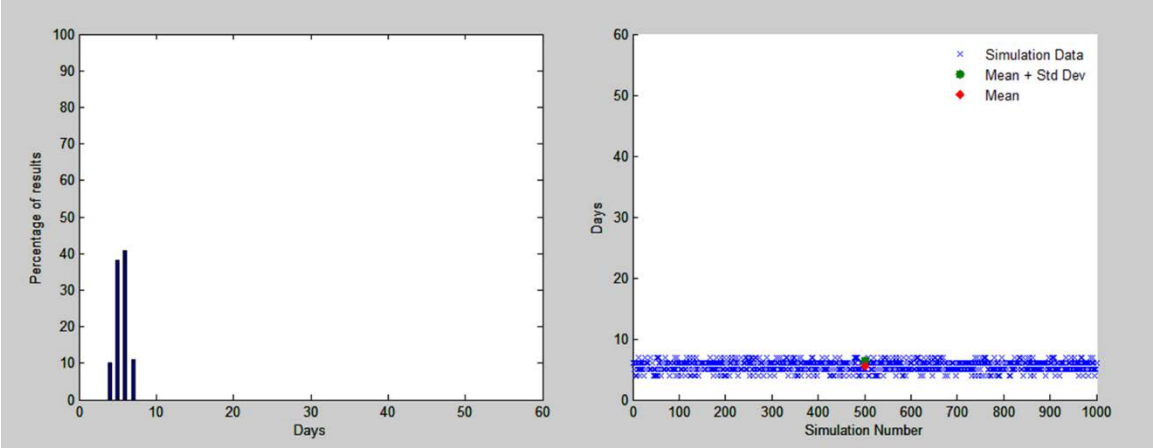


Figure 18: End Date Histogram (Left) and Scatter (Right) Plots for Figure 17

Feedback loop verification was the next to implement. Taking the baseline process (with no task completion distribution), a feedback loop (Figure 19) with a 20% chance of iterating was added. The feedback loop should represent a geometric series across three tasks. The deterministic three days should have an 80% likelihood, with a 16% likelihood of six days, a 3.2% likelihood of nine days, and so on. The average should be (1/.8)*3 days, or 3.75 days.

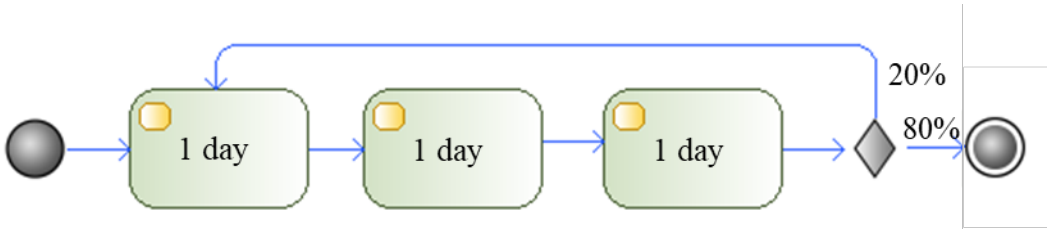


Figure 19: Process With Feedback, no Task Distributions

However, the results were always three days. The original code assumed that completing all tasks accomplished the objective, so even when the process would queue task B, the process would be complete and the code would end. This failure was solved by two methods; first, the assumption was updated to ensure that no queued work

remained, and secondly a null task with the lowest priority can be added to ensure that all predecessors have completed before the process completion (Figure 20).

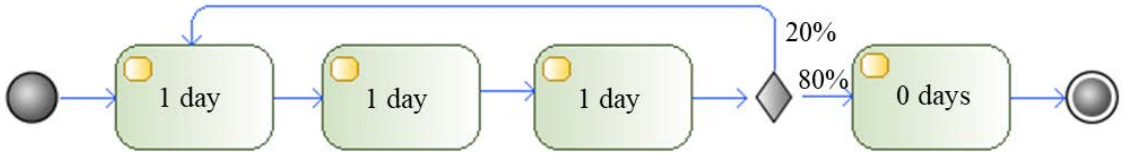


Figure 20: Corrected Process With Feedback, no Task Distributions

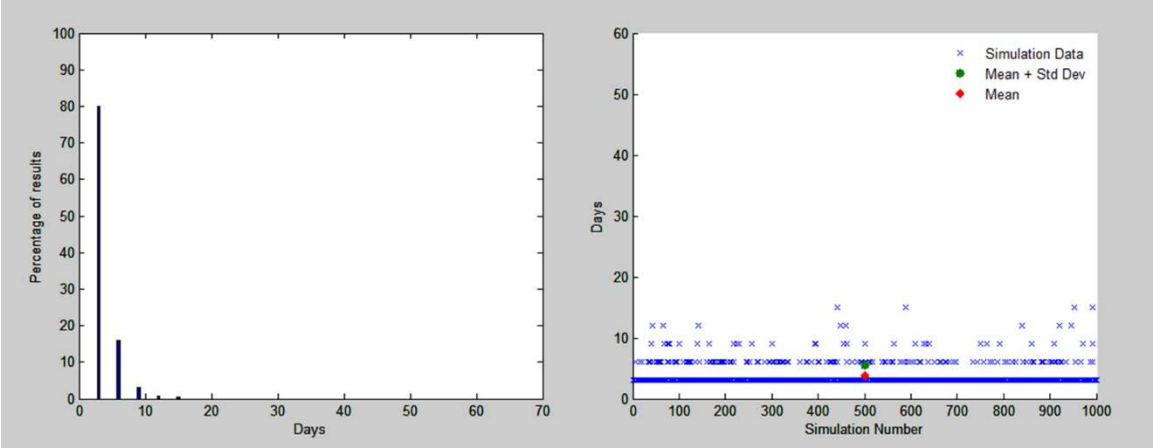


Figure 21: End Date Histogram (Left) and Scatter (Right) Plots for Figure 20

By combining both feedback and task distributions into the simulation, a bell distribution around each iteration should be seen. Taking the process from Figure 20 and adding a random value between -0.5 and +0.5 days to each task, a distribution should start showing around day 4.5, 9, 13.5, etc.

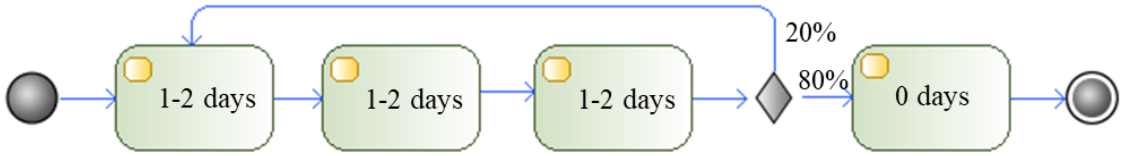


Figure 22: Process With Feedback and Task Distributions

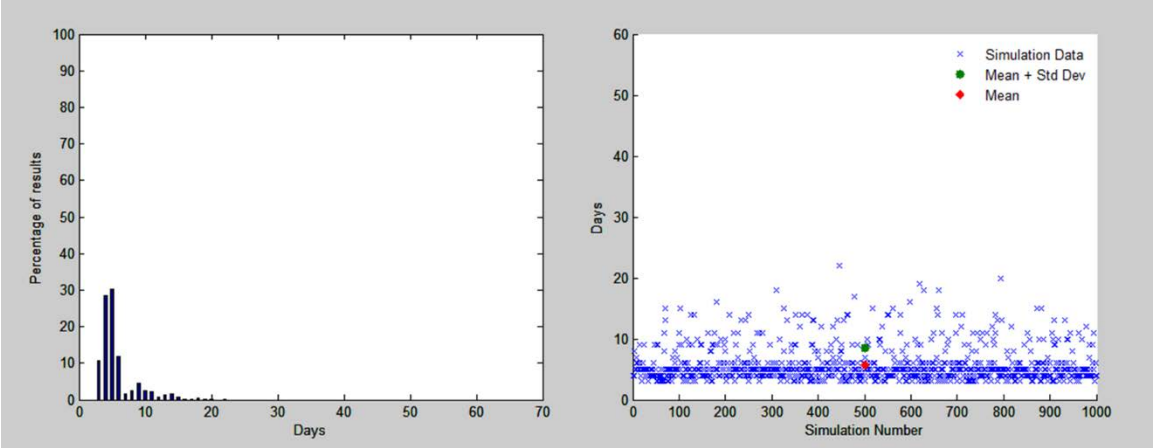


Figure 23: End Date Histogram (Left) and Scatter (Right) Plots for Figure 22

The recorded results are exactly what was expected for the sample size. Approximately 80% of results centered around 4.5 (the case where feedback never occurs), 10% chance for a minimum time (50%^3 for each task taking only 1 day, and a 80% chance to not iterate = 10%), another distribution around 9, and small, non-representative samples beyond 12.

Effects from the feedback and distributions were able to overlap, and visually verifying that the tool was functioning properly was difficult. To produce a more visually representative verification, a more extreme example was formed. Increasing each tasks to five to eight days, the random distribution around each mean increased, and the feedback loop separated out further than the distribution. Figure 25 shows wider distribution ranges with means separated further apart, and the scale of the histogram is improved.

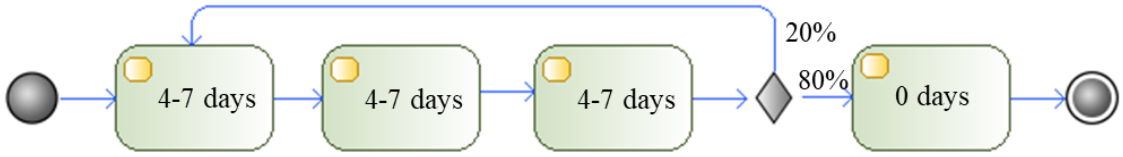


Figure 24: Feedback and Significantly Large Task Distributions

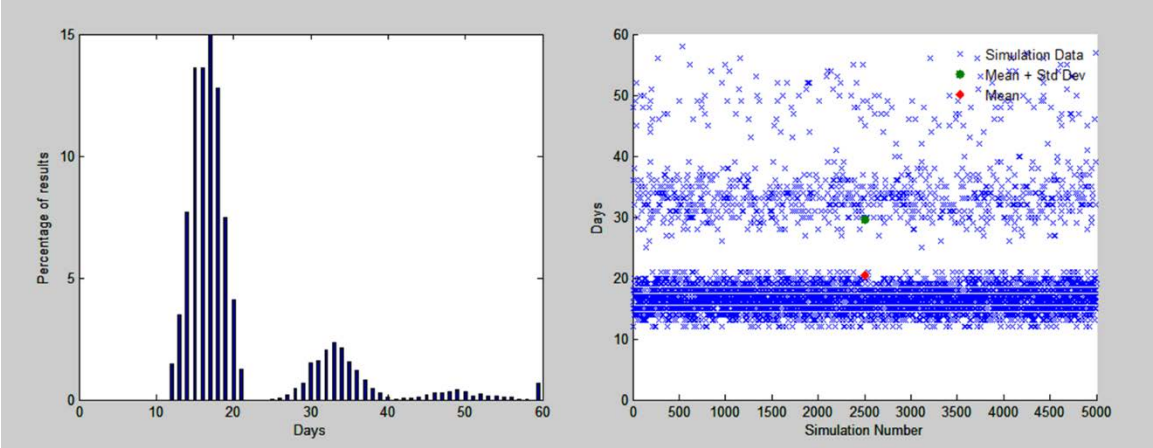


Figure 25: End Date Histogram (Left) and Scatter (Right) Plots for Figure 24

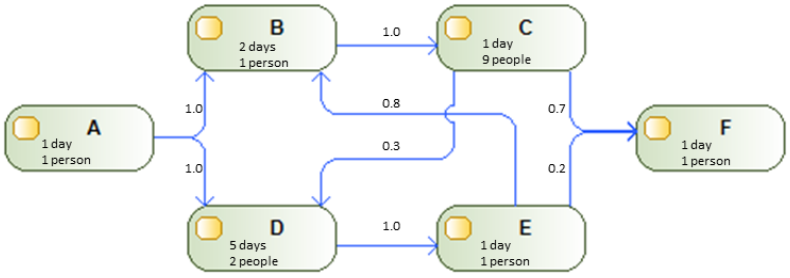
Validation of the Process Evaluator Using the 6-Step Example

To validate the functionality of the whole tool, return to one of the less intuitive statements from Chapter II. It was stated that the same process implemented in different ways can evaluate differently. One of these is how to prioritize limited resources against a process that has multiple valid means of completing the tasks. By establishing COAs on how to apply resources to tasks, the process evaluator should be able to find differences. The example should be able to be explained logically without simulation, and verified with simulation.

The process that will be used to demonstrate this principle will be the example in Figure 8. This example has complex feedback loops, and is difficult to calculate expected

time and variance (which is only caused by feedback loops, task time is deterministic). The expected mean should be partially represented by the standard and easily calculated case; no iteration was incurred. However, even though the feedback loops are the same for any prioritization, one should evaluate more favorably than the others.

By varying only the task priority, the comparison should highlight superior and inferior methods of prioritizing the tasks. Because A and F are the first and last tasks in the process, respectively, only the four tasks in the middle can be reprioritized in the DSM. While there are twenty-four potential prioritization permutations, feasibility criteria dictates that the second task would need to be either B or D, leaving six feasible permutations.



%	A	B	D	C	E	F	d	p	t
A		1	1				1	1	1
B				1			2	1	2
D					1		5	2	2.5
C			0.3			0.7	1	9	0.1
E		0.8				0.2	1	1	1
F							1	1	1

Figure 26: 6-Step Diagram and DSM

Simulating this process across the six feasible permutations should return expected results. By simulating with the constraint that the process only has two workers, the prioritization of what tasks the workers should do will provide different results. To depict how the process will execute before considering reiteration, a visual representation of all six COAs and how the two workers execute tasks has been drawn below (Figure 27).

Each COA was evaluated for the hypothesis that the case that no additional iteration is incurred will roughly approximate the iterative results. With each of the task's blocks to scale and placed where they would start, and only two works, it should be expected that ADBECF and ADEBCF should provide the best results, and ADBCEF should give the worst results. The other 3 orderings should give similar results.

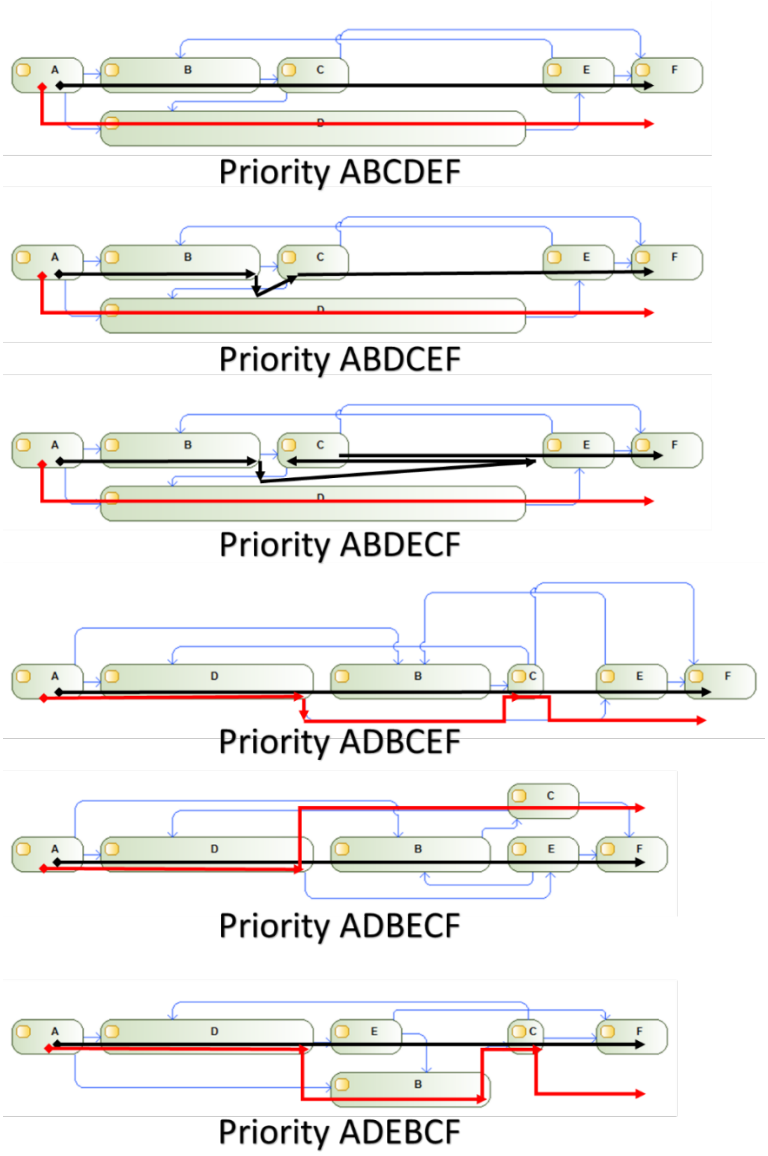
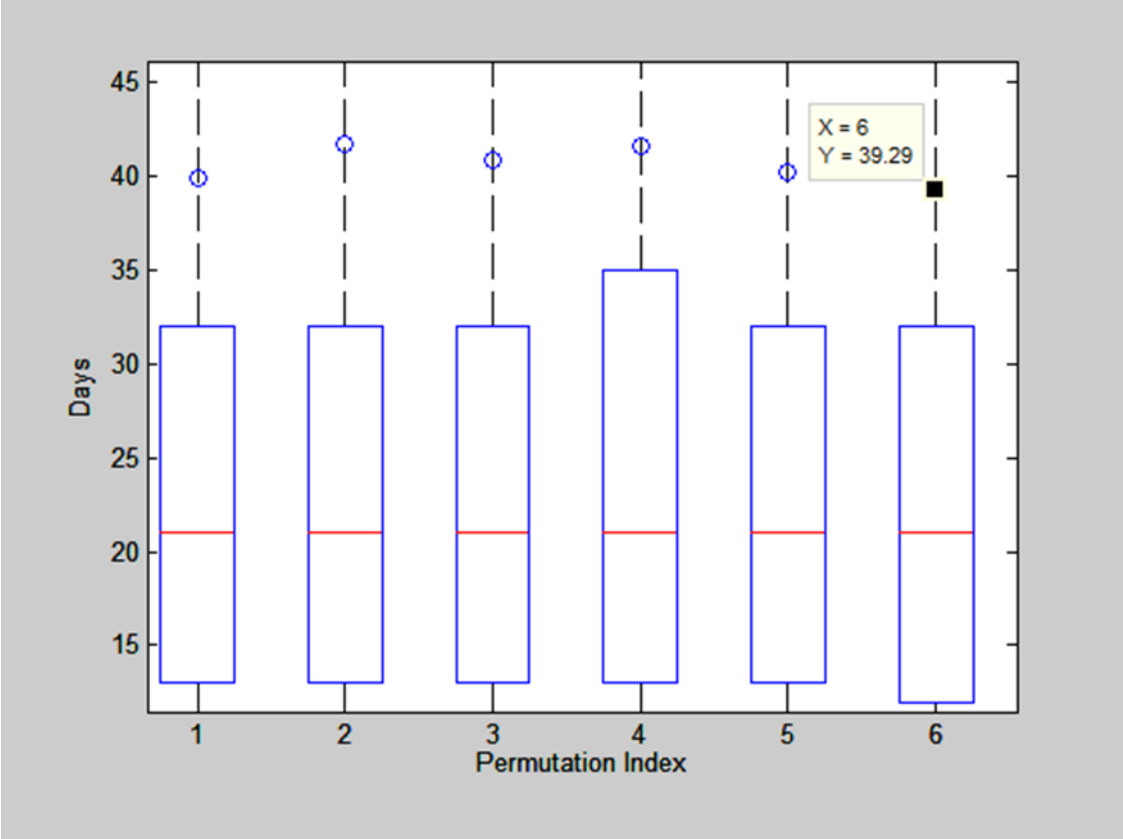


Figure 27: 6 Feasible Prioritizations & Worker Loading, Tasks to Scale



A	A	A	A	A	A
B	B	B	D	D	D
C	D	D	B	B	E
D	C	E	C	E	B
E	E	C	E	C	C
F	F	F	F	F	F

Figure 28: Performance Data for 6 Step COAs

Each of the six COAs was simulated five-thousand times each, with a limitation of two workers. The best case found was one of the two theorized best cases, and the worst case was the expected worst case. In addition, two unusual results are found.

First, the three execution orders starting with “AB” should have been identical, and while they had the same median result (red line internal to the box plot), they had different means and standard deviation (summed as the blue circles). The second

discrepancy is that ADBECF is worse than ADEBCF. These discrepancies are due to the business execution rules allowing an extra half day of work in some cases (ie if rework was queued before task C begins in COA 6, COA 6 becomes better than COA 5). Additionally, the iteration causes large amounts of variance, and the sample size was too small to fully account for random sampling.

Use of simulation was able to verify that the prioritization of tasks can have statistical differences and identify an optimal choice. Simulation was also able to account for iteration, and quantify how much each implementation was impacted by the stochastic nature of the process.

Experimental Procedures

As described in the research methodology section of this chapter, the first step of experimentation will be to develop models of existing processes, and analyze them in the Process Evaluation Tool. Attempts to analyze and improve the established processes will help identify the limitations of the Process Evaluation Tool to aid DMAIC process improvement. Limitations for simulating process improvement will be used to scope what types of process improvement case studies can be evaluated during the second step of experimentation.

The second portion of experimentation will be to find existing case studies where the author measured data, attempted to improve a process, and provided the result. The case study's process will be replicated, simulated, and the model prediction will be compared to actual results. It is expected that this will establish a rough understanding of the accuracy of the tool. The case studies will need to have process and data

documentation sufficient to produce a model. Case studies where process improvement failed to achieve improvement goals would be ideal, as the Process Evaluation Tool should be able to indicate that flaws in assumptions or corrective actions exist.

It is expected that the tool should not always produce accurate results, and the failures to correctly predicted results will be analyzed. This analysis should determine one or more "pitfalls" towards Modeling and Simulation in DMAIC. It may also show critical data that need to be collected in the "measure" step beyond what would be considered normal data in unbounded process improvement.

Finally, the Process Evaluation Tool will be used against real-world process improvement events. The tool will be used to explore process improvement options, recommend implementations, and show results (as available at the time of this paper's conclusion).

Description of Dependent and Independent Variables.

Process improvement by its nature supports change. In spite of change, the definition of a process implies that the objective of a process can not change. If the improvements to a process no longer achieve the objective, the process has become invalid. Assumptions on process feasibility include that all mandatory steps have been completed at least one time after preceding mandatory relationships were followed. The evaluation criteria is also assumed to be static, as the improvement goals should be set prior to M&S in DMAIC. The process prior to improvement is also not a variable.

Resources, constraints, the process's non-mandatory tasks, the non-mandatory relationships, and prioritization of tasks are all independent variables. Whenever reasonable, the resources to complete the process will be held constant.

The dependent variable is set by the evaluation criteria for a given process improvement problem. The default improvement metric for this paper is the sum of mean plus variance of time to complete.

During case study replication, the process before improvement and the improvement measures will be independent variables.

Experimental Tasks

Experimentation is done by modeling and simulating processes. Variation of the inputs is performed either on the process model itself (to include resources and constraints), or on the execution order/priority of the process's tasks. Evaluation is compared across multiple COAs the most favorable process improvement for the evaluation criteria. Analysis of modeling and simulation failures will determine boundaries and limitations.

Experimental Equipment

The study will be conducted via Matlab 2014. The computer running the software is a MSI CX61 laptop with no upgrades, specifications of CPU running with a 2.6 GHz dual core processor and 8 GB RAM.

Assumptions

When a process changes, the performance of the system may change. These changes involve a learning curve due to the change, the steady state efficiencies gained or

lost, and the efficiencies gained or lost by the factors external to the process. The process model would—in theory—include such factors as part of the model.

At times the factors were not present under the old process. By changing the process, addition or removal of tasks may be necessary to meet the objective of the process. Automatic calculation of COAs can not accurately predict new factors, and the exclusion as part of the experimentation is required. It is assumed that the manager trying to improve a process would be able to determine that a given solution is impractical.

Analyses Method

Experimentation will use Monte Carlo simulation to calculate performance for given process, and evaluate through the Process Evaluation Tool developed earlier in this Chapter. By applying the evaluation across multiple process improvement COAs, comparing evaluation results will identify the COA that evaluated best.

When experimenting with case studies, evaluation of the single "truth" point is difficult to compare with a Monte Carlo simulation. A P value formula is not meaningful if a process has a non-Gaussian distribution. A comparison between the point value from the case study and a large-sample of Monte Carlo results will allow a rough view of if the single result was probable for the model. The comparison between simulation and the result will only subjectively determine if the model is representative of the process.

Summary

Simulating a given process's steps, interactions, resources, constraints, and work prioritization against evaluation criteria has been translated into a tool. The Process Evaluation Tool provides a statistical representation of expected performance for one or

more COAs of a process. The tool has demonstrated the ability to verify or disprove hypothesizes on process model performance following the implementation of improvement options.

The Process Evaluation Tool that was developed will be used experimentally to find how it can aid improving a process. It is assumed to help verify if a process COA should achieve the improvement goal, and the accuracy of the prediction will be viewed. Limitations of the tool will determine heuristics in simulating process improvement, and real world application will guide use of the tool.

IV. Analysis and Results

Chapter Overview

This chapter applies the Process Evaluation Tool to processes to learn limitations, accuracy, heuristics, and recommended approach. It steps through these experiments to determine how to evaluate the performance of a process following improvement measures that have not yet been taken, and how to use the evaluation results to improve a process reliably.

Experimentation by Replicating Established Processes: Insurance Claims

To start with a simple example of how the tool can be used, a process with no iterative relationships will be improved using the DMAIC methodology. It is beneficial for this process to be generic and well known to establish a common understanding of the tasks and their impacts. Arbitrarily chosen, Geico's insurance claims description from their support center was used (Learn About The Complete Insurance Claims Process, 2016). Note that this is not Geico's implemented process, but a model based on their website description of the process.

First, the description of Geico's insurance claim process was converted to a digraph (Figure 29) and corresponding DSM (Figure 30). Where specific data was missing, "best guess" assumptions were made about the model through prior experience with insurance claims. Data was arbitrarily generated for the time, cost, and variability of these steps as well as the likelihood to follow any specific relationship following a split.

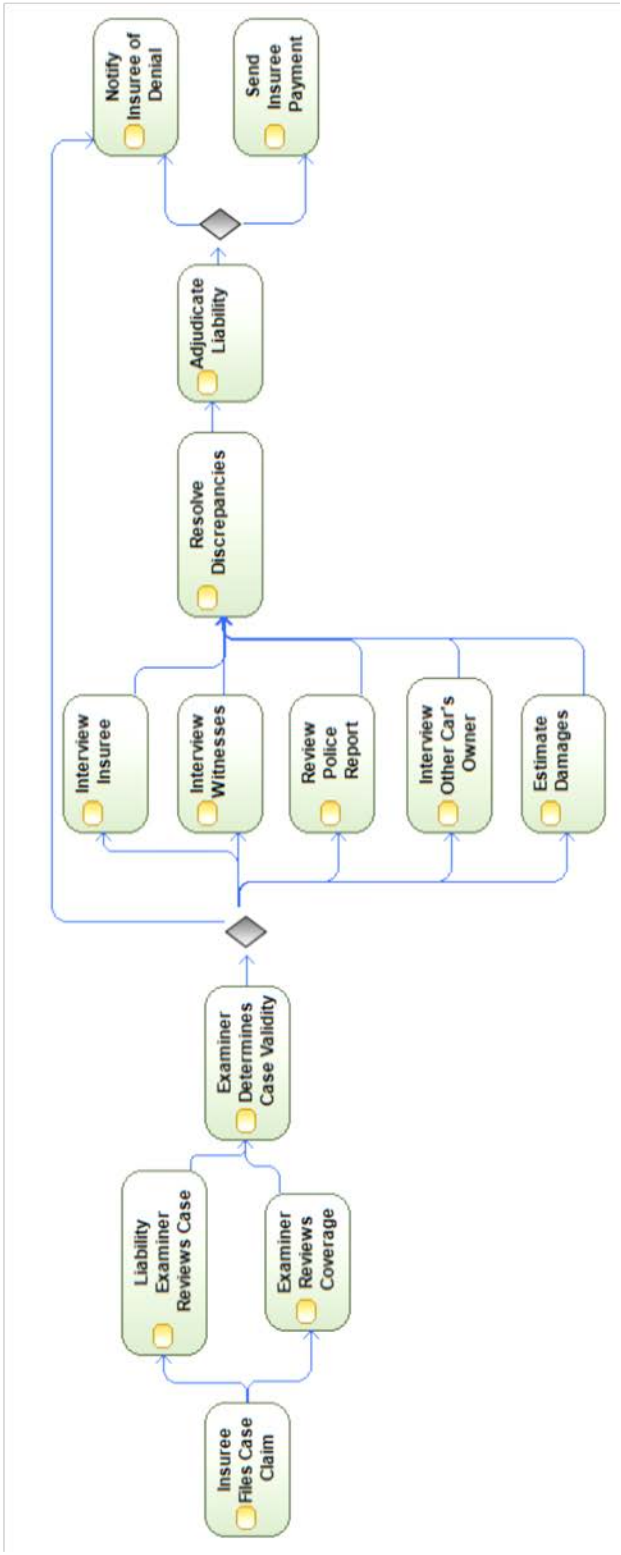


Figure 29: Insurance Claims Process Digraph

In this example, an insurance claim evaluator follows a typical large-company claims process. Presumably, one evaluator handles the entire case. Several tasks need to happen, sometimes concurrently and sometimes linearly. The sole evaluator can only be making progress on one step at a time with the exception of the damage estimate, which is handled by a second source. There are three exit conditions; Payment for the claim, Denial based on not being covered for the type of incident, and Denial based on liability.

	Insuree Files Claim	Review Case	Review Coverage	Determine Validity	Interview Insuree	Interview Witness	Review Police Report	Interview Other Car's Owner	Estimate Damages	Resolve Discrepancies	Adjudicate Liability	Send Payment	Deny Claim	**Element ID**	Time to Perform	Time Variance	Max Workers	Employee Type	Cost to Perform	Cost Variance
Insuree Files Claim		1	1											1	0	0	1	1	0	0
Review Case				2										2	1	1	1	1	15	15
Review Coverage				3										3	1	1	1	1	15	15
Determine Validity					5	5	5	5	5				4	4	1	0	1	1	15	0
Interview Insuree										6				5	1	1	1	1	15	15
Interview Witness										7				6	2	2	1	1	30	30
Review Police Report										8				7	3	1	1	1	45	15
Interview Other Car's Owner										9				8	2	2	1	1	30	30
Estimate Damages										10				9	8	8	1	2	350	150
Resolve Discrepancies											11			10	4	3	1	1	60	45
Adjudicate Liability												12	13	11	3	0	1	1	45	0
Send Payment														12	1	0	1	1	7515	5015
Deny Claim														13	1	0	1	1	15	0

Figure 30: NDSM for Insurance Claims Process

Applying DMAIC, the first step is to define the goals. As a simple and common goal, the “Define” objective is to reduce the average time the process takes and the

variance within the time as close to zero as possible. The next step would be to "Measure" the process data, but the data was arbitrarily fabricated to facilitate the example. For this experiment, time is measured in hours, and cost is measured in dollars.

The next step is to "Analyze" the process, and start brainstorming. Analyzing the process diagram (Figure 29) there is a "fast" path by denying the claim based on coverage. Time and variance could be significantly reduced by making this the most likely outcome of the process. It should be assumed that the company is denying the correct number of claims to maximize business.

Another process diagram issue is that "Estimate Damages" does not seem to have accurate relationships. It is reasonable to assume that the cost estimation is only required to know how much to pay if the process ends on payment. Because the relationship is not required for each outcome, it is not mandatory for validity, and can be re-linked if there is benefit to doing so.

Switching analysis to the NDSM data, the largest value generated in the "Time" data is the estimation of the damages element. It has the longest duration, and the largest range (in this case, four to twelve hours evenly distributed around a mean of 8). If the damage estimate started when the claim was filed, the end time should improve. Notably, this is a wasteful "improvement," but would be a possibility for only improving time with no consideration for cost.

A second target in analyzing the NDSM is the large amount of concurrent work following the decision that the claim is valid. After determining validity of the claim, the DSM puts four elements on the same resource (worker type one, the liability examiner) at

one time, creating a bottleneck. Several options exists to reduce that bottleneck, but the following two will be evaluated.

The first option is to add a second liability examiner to the interviews and "resolve discrepancies" tasks. The other options is to restructure the interview/review process. Four sources are taken in, then checked for discrepancies, then passes to adjudication. If three sources align (especially both parties for a car crash) and there were no discrepancies, reviewing the fourth source would not make sense. By modifying the process to review three sources, then check for discrepancies, then there is a chance that sources agree and reviewing the police report can be bypassed. If that chance does not happen, the police report is used in conjunction with the discrepancies to adjudicate liability, as seen in Figure 31. For one worker, this should mean there is a chance the police report is bypassed without adding additional work.

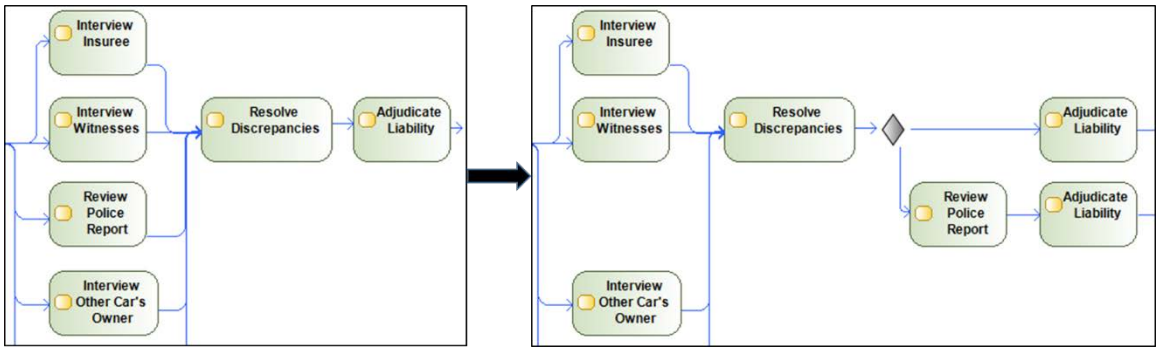


Figure 31: Restructuring Reviews

Another way to perform analysis is to simulate the baseline process, and review the process execution data. Simulating the end date (Figure 32), it can be seen that the 30% likelihood to deny the claim early drives standard deviation to be nearly 10 and significantly larger than it should be from the bell shaped distribution centered on 24.

Increased deviation reduces the desirability, and should be a target for improvement. It is not logical to eliminate a quick and easy claims denial in the real world, but it does stand to reason that eliminating the early completion dates drop the variance at the cost of increase the mean.

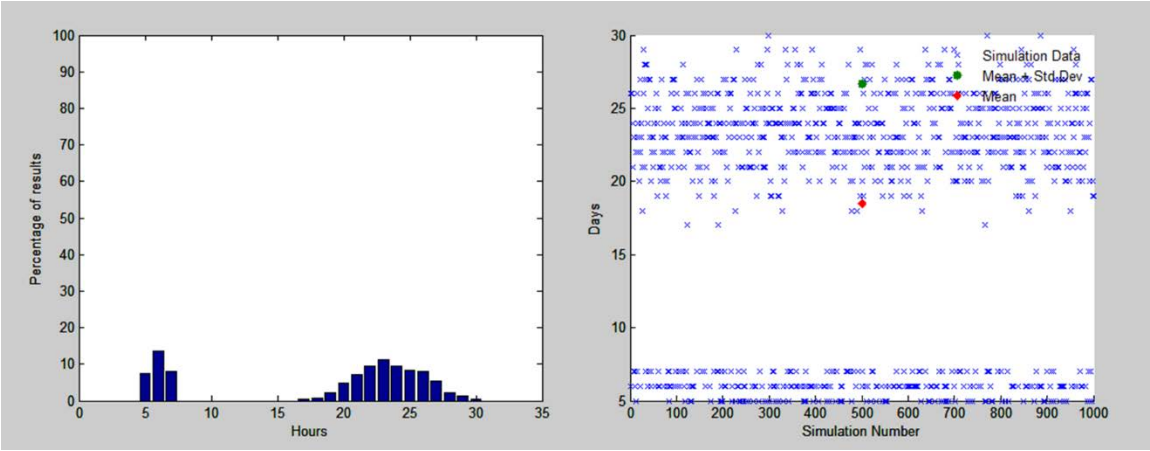


Figure 32: Baseline End Time for Insurance Process

In addition to reviewing the end dates from simulation, the work schedule was reviewed for when an individual element was likely to be in work (Figure 33). It can be seen that as time progresses, the stochastic range of each task is passed to subsequent elements and compounds. In particular, the long and uncertain execution time of "Estimate Damage" forces subsequent tasks to have a wide distribution of end dates.

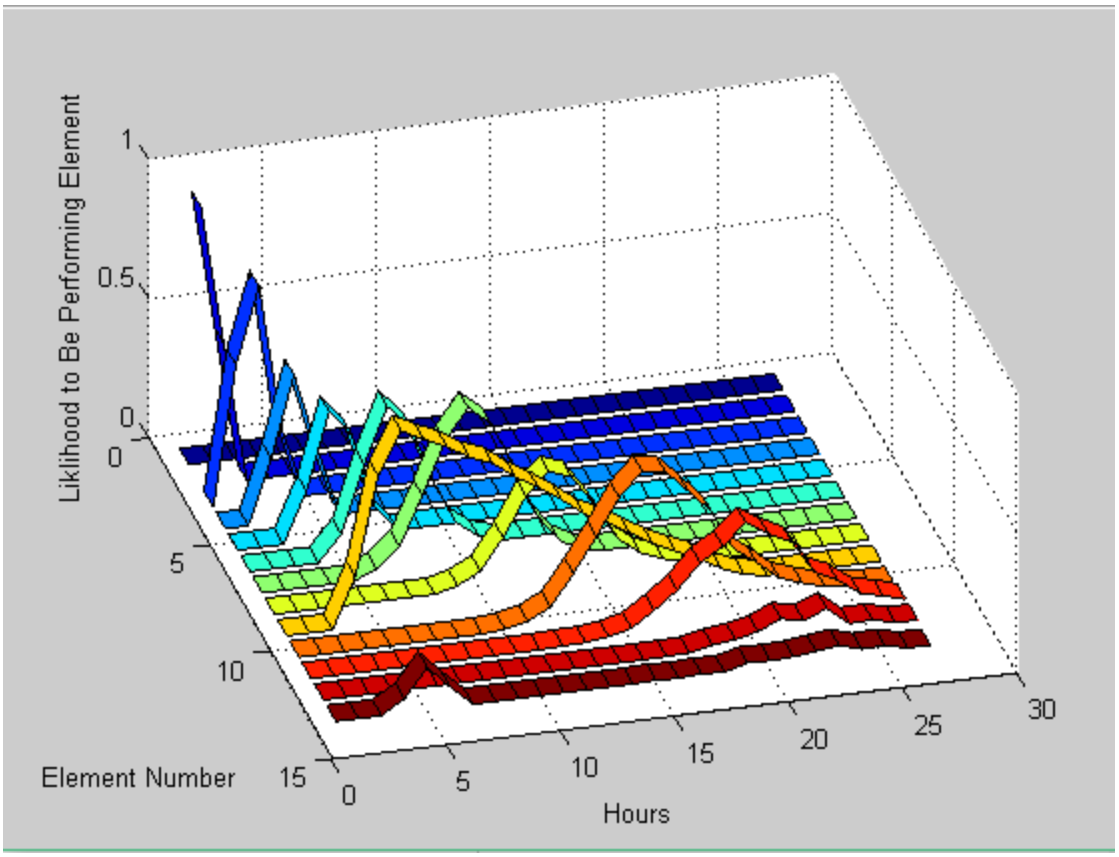


Figure 33: Probability for Element Y to be Performed at Time X

The final portion of analysis is varying the execution priority of the process. The process itself has thirteen elements, and therefore has 6,227,020,800 permutations of the prioritization. Attempting to reduce the permutations to only what was feasible by computer caused memory overflows.

A method to find feasible prioritizations would be to focus on where forks and merges were, and prioritize internal to the "isolated" portions of the process. Focusing only on elements one through four; there are only two feasible execution orders for those elements; 1, 2, 3, 4 and 1, 3, 2, 4. One must precede two and three, and four must follow both.

Elements five through eight can be reprioritized without impacting validity, with 24 feasible orders. Element nine has its own manpower, and will not be interrupted by the liability examiner's priority.

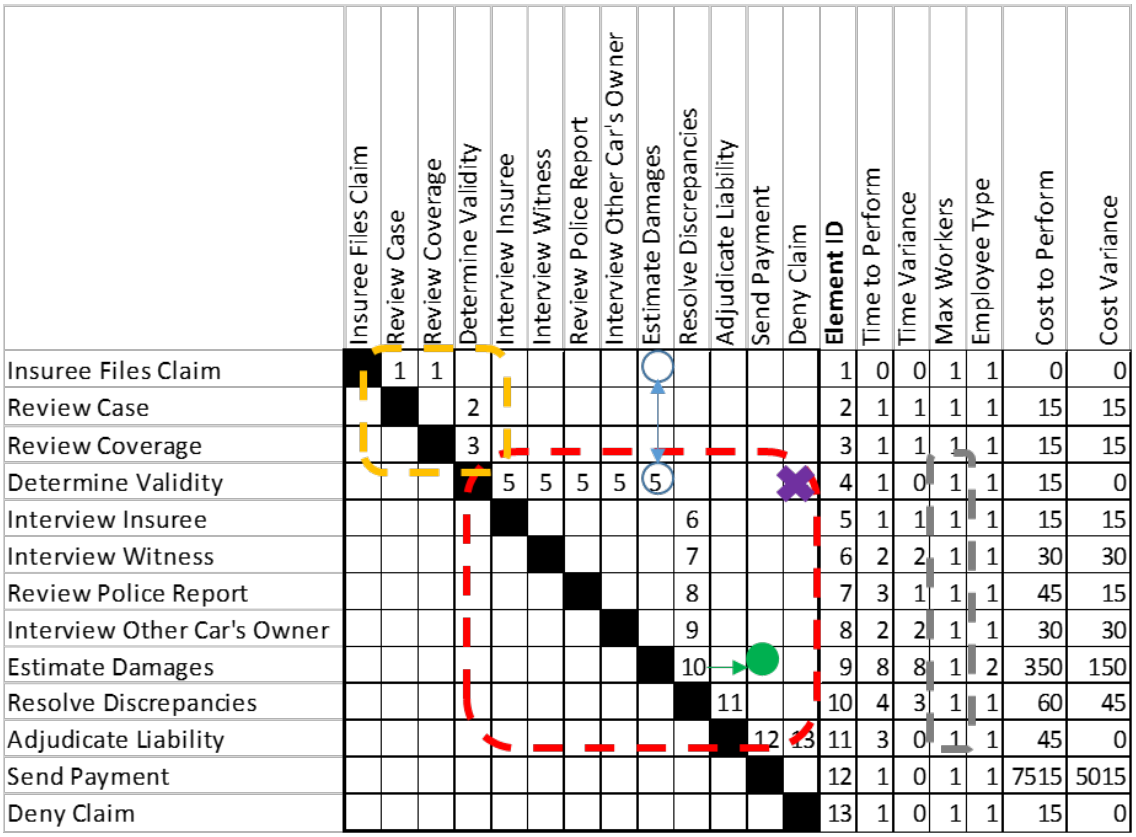
Elements ten and eleven only have one feasible order, and elements twelve and thirteen can not both execute. Because denying the claim in two different outcomes was implemented in the DSM as one element, prioritization of denying the claim must come before element six or the entire process will execute prior to denying the claim early. Other orderings would not be feasible for denial.

The four major sections (elements one through four, five though nine, ten and eleven, twelve and thirteen) will only execute in order or jump from element four to thirteen. While there are six billion permutations to pick from, bounding the options to feasible process paths, there are only forty-eight permutations. However, one worker is performing each of the tasks that can be reprioritized, and it should be expected that there is no change to performance. One reprioritization will be evaluated against the initial priority to verify this theory.

During analysis, six theories of how to improve the process have been developed. Each theory has two options; change the process or do not. Because the combination of theories can be made, a total of 2*2*2*2*2*2 = 64 options exist for evaluation. A subset of the options were selected as COAs for evaluation. The Process Evaluation Tool was first passed the baseline process, each of six COAs individually, and all six COAs at the same time.

Each improvement can be viewed in Figure 34, with the changes to the DSM, resources, relation changes, and prioritization impacts colorized. Moving the estimation

of damages' outgoing relationship to "send payment" should buy more time, giving two options (green). Moving "estimate damage" to the start of the claim should reduce overall time, giving two options (blue). Using a liability examiner presents two options (gray). Editing the review/interview process to require a minimum of three sources if they agree gives five options (red), but the longest element (Review Police Report) will be excluded from simulation as the "best case" of the five. Eliminating the early ability to deny the claim eliminates one of the three process exits (purple). Prioritization analysis on the first four elements gives two options (orange).



	Insuree Files Claim	Review Case	Review Coverage	Determine Validity	Interview Insuree	Interview Witness	Review Police Report	Interview Other Car's Owner	Estimate Damages	Resolve Discrepancies	Adjudicate Liability	Send Payment	Deny Claim	Element ID	Time to Perform	Time Variance	Max Workers	Employee Type	Cost to Perform	Cost Variance
Insuree Files Claim		1	1											1	0	0	1	1	0	0
Review Case				2										2	1	1	1	1	15	15
Review Coverage				3										3	1	1	1	1	15	15
Determine Validity					5	5	5	5	5					4	1	0	1	1	15	0
Interview Insuree										6				5	1	1	1	1	15	15
Interview Witness										7				6	2	2	1	1	30	30
Review Police Report										8				7	3	1	1	1	45	15
Interview Other Car's Owner										9				8	2	2	1	1	30	30
Estimate Damages										10				9	8	8	1	2	350	150
Resolve Discrepancies											11			10	4	3	1	1	60	45
Adjudicate Liability												12	13	11	3	0	1	1	45	0
Send Payment														12	1	0	1	1	7515	5015
Deny Claim														13	1	0	1	1	15	0

Figure 34: Color Coded NDSM Edits for Potential Improvements

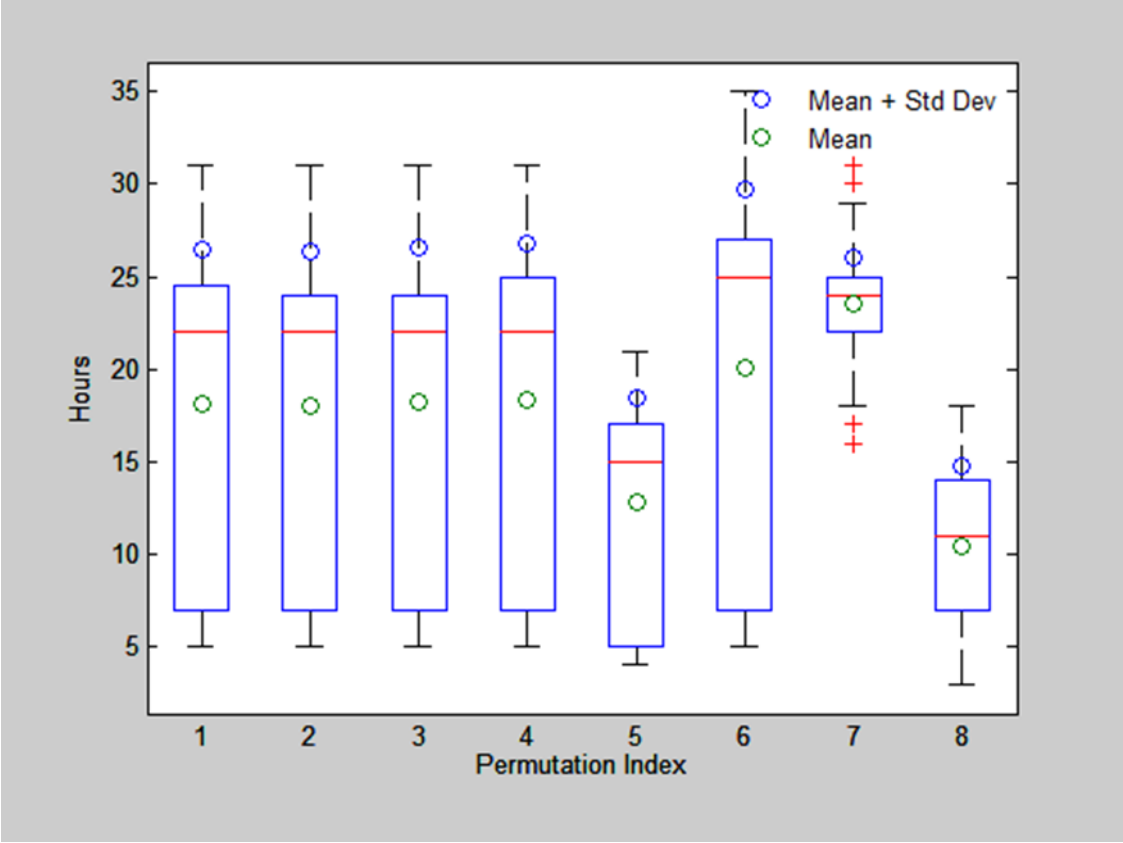


Figure 35: Simulated Insurance Claims Data

Figure 35 lists the box plot, mean, and evaluation value for each of the following COAs:

1. Baseline Process
2. Reprioritizing the process (Yellow)
3. Changing the "Estimate Damages" relationship (Green)
4. Start "Estimate Payment" as soon as the process starts (Blue)
5. Add a second liability examiner to the manpower pool for the process (Grey)
6. Restructure the reviews as depicted in Figure 31 (Red)
7. Eliminate the 30% chance for early claims denial (Purple)
8. Include all six improvement theory changes

Most of the results are intuitive. Changes to the prioritization in the first four steps had almost no impact. Changing the linking from damage estimation had almost no impact. Estimating the damages earlier had almost no impact. Increasing manpower had a significant positive effect. Eliminating the early end significantly reduced standard deviation, but the mean value increased more than the standard deviation was reduced.

The restructure of the bottleneck evaluated poorly, which was unexpected. The redesign of the reviews should have only had a potential for positive effects. Reviewing the data, it was found that the NDSM had not been updated to correctly reflect the intent of Figure 31. A relationship from “Review Police Report” to “Resolve Discrepancies” existed, adding one additional iteration 75% of the time. The average result was 3.375 additional hours of work on the average case to remove 0.875 hours of work, increasing the mean by 2.5 hours due to a assumed relationship. The data recoded suggests about 1.8 additional hours of work, on average.

Applying all options gave the best performance, but both positive and negative performing COAs are included. Given several no/minimal-impact changes to the system, two bad changes, and one positive change, applying every change should have worse performance than simply applying the one successful change. In a stochastic context, the implication is that there may be a two-or-more-factor interaction by applying multiple changes at once.

Two additional COA evaluations were added. First, simply including the three options that showed any improvement were simulated, applying changes from COAs two, three, and five (results on index 9 in Figure 36). The tenth COA involved applying all changes except COA seven (result on index 10).

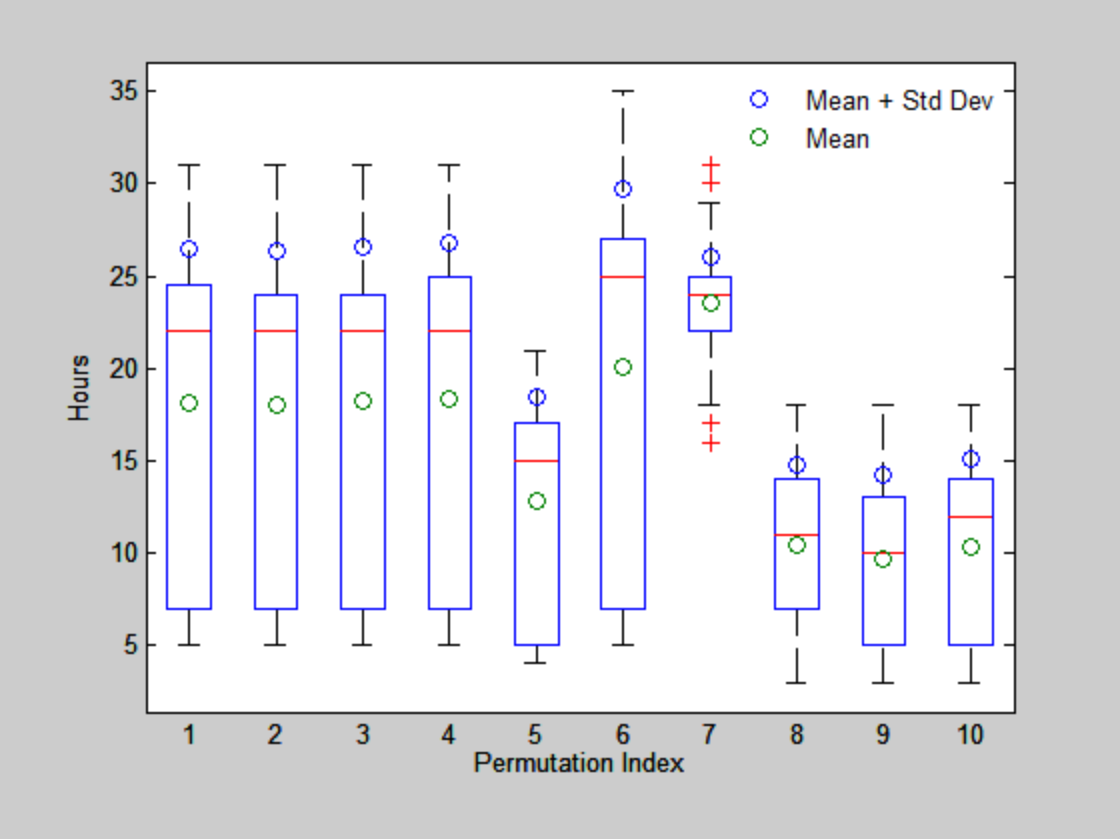


Figure 36: Additional Evaluations of Insurance Claims Data

The Process Evaluation Tool has been demonstrated to be useful in evaluating the benefit of resource changes, process relationship changes, and task changes. It has also demonstrated that it can find failed assumptions during restructuring elements within the process.

It is uncertain at this point if prioritization has been demonstrated in this example, as improvements were not notably seen when changing only prioritization. The ability for difference under this assumption should not have been likely with only one worker. In addition, positive results where priority was changed were aliased with the addition of a second worker.

Replication of Case Studies: Improving Hospital Bed Availability

To be confident that the Process Evaluation Tool is able to accurately forecast if a proposed COA is the ideal improvement solution, a more complex analysis of its accuracy is required. To achieve this, a process improvement case study will be replicated, and analyzed for accuracy.

A case study with stochastic data of the initial process, improved process, and a list of improvement measures is desired. Additionally, replicating a result more complex than “it got better” or “it didn’t improve” would provide a better assessment of the Process Evaluation Tool. The case study selected is a time reduction study with significant stochastic issues, which were notably improved, but fell significantly short on objective goals.

Rosalie Sager and Eric Ling (Sager & Ling, 2017) applied Six Sigma to a medical center’s process to increase bed availability. The issue was that when patients were given a discharge order, the process took significantly longer than desired (155 minutes) or allowable (235 minutes) to make the bed available for the next patient. The average was 271 minutes to make the bed available, and the process performed worse than the max allowable limit 60% of the time.

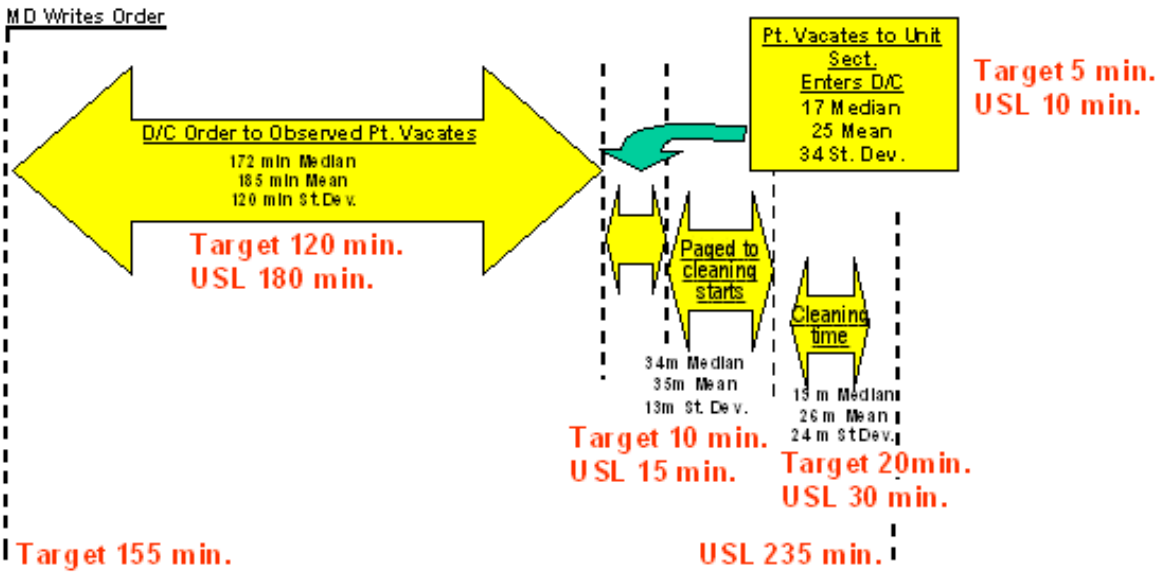


Figure 37: Timeline Data for Bed Availability (Sager & Ling, 2017)

Sager and Ling defined the improvement objective as a 155 minute average to execute the full turnover. By measurement and analysis, they determined that the primary issue was an unpredictable range that the process would complete. The likelihood that the average would be executed was statistically unrealistic, and the spread of values was so wide that the timeline was unpredictable and too long.

To see how accurate simulation predictions can be, the baseline process is replicated in an NDSM model (Figure 38) and run in the Process Evaluation Tool developed in Chapter 3. An element to initialize the process and an element to conclude the process were added, but with no evaluation criteria. The same implement steps are applied to the model to predict how well the five process improvement changes will increase the performance metric of average time to complete the process.

	Write Discharge	Patient Prep	Idle Time	Housekeeping Arives	Cleaning	Room ready	Element ID	Time to Perform	Time Varience	Max Workers	Employee Type
Write Discharge		1					1	0	0	1	1
Patient Prep & Departure			2				2	185	205	1	1
Idle Time to Paging Housekeeping				3			3	25	50	1	1
Housekeeping Arives					4		4	35	23	1	1
Cleaning						5	5	26	45	1	1
Room ready							6	0	0	1	1

Figure 38: Baseline DSM for Bed Availability

While the sample point data was unavailable for the initial process, Ling and Sager provided mean, median, and standard deviation for each task before implementing any changes. One shortfall of the provided data is that a non-Gaussian bell curve is implied (as the mean and median do not correspond), but not available.

Building a model of processes, 6 steps were generated; an element for starting the process, the four steps listed in Figure 37, and an element to finalize the process. The statistical nature of the tasks needed to be approximated for each step to replicate the standard deviation, as the specific curve data was not provided and was known to be a non-bell shape.

Simulated results had mean values working fully. Standard deviation was applied by assuming square distribution. The task where the unit secretary would notify housekeeping was not able to replicate standard deviation, as a negative value could occur assuming a square distribution. Deviation had to be capped at plus or minus 100%

of the mean value to prevent negative values in simulation time. The notify task was simulated with standard deviation of 25, 36% less than measured data.

As Ling and Sager has reported, simulated data saw issues with stochastic results. The left graph of Figure 39 shows that no specific time is more than 3% likely, and attempting to find an 80% confident guess at when the bed would become availability requires a 6 hours of uncertainty.

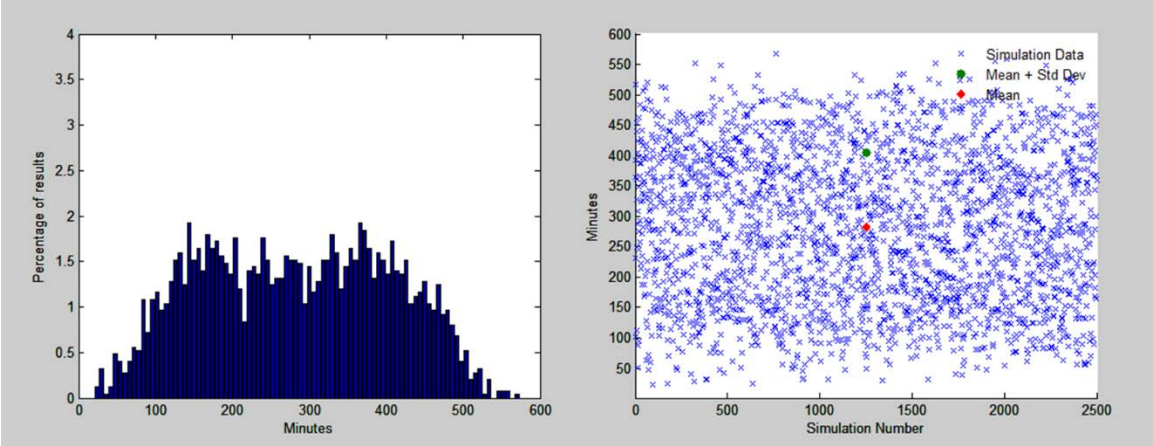


Figure 39: Baseline Time for Bed Availability

To bring control to their process, Ling and Sager implemented five changes to the process to improve performance. The first improvement was that special status was given to patients who received discharge orders, but had additional testing in the hospital required, or needed to be provided special equipment. Next, the staff was trained about the bed tracking system, and aware of expected handoffs in the bed turnover process. Third, the patient was asked prior to the discharge order how they were getting home, to reduce unexpected delays. Patients were also provided a video explaining the discharge process, so they would more effectively communicate their departure. Finally, paperwork the patient required before leaving began being filled out prior to issuing the discharge.

Ling and Sager did not provide why the measures were implemented, where they impact the model, or to what degree the improvement is anticipated. Each improvement was reviewed for which of the four tasks it was likely to improve, and then each was given an estimated improvement results.

Placing priority status on patients who had received a discharge does not change *when* the patient will leave, but the staff's expectation of when they would leave; this improvement reduces the idle time after the patient leaves to when the housekeeping staff arrives to clean. Educating the staff on the bed tracking system also targeted the idle time interjected by the staff. Verifying the transportation method allowed for two impacts; the staff could more appropriately expect when the patient would leave, but also targeted the patient's potential delay while making transportation arrangements and waiting for transportation to arrive. Providing a discharge video would improve departure time for the patient, and would reduce the idle time in the unit secretary realizing that the bed had been vacated. Finally, doing paperwork early helped reduce the time the patient required to have all items ready to leave the hospital.

An estimate for expected improvements due to the five changes was not available. I am not a subject matter expert on hospital operations, so I applied a 20% reduction to both mean and standard deviation each changes' respective step/s. There was one exception, where the completion of paperwork prior to discharge was assumed to be a static reduction of 5 minutes.

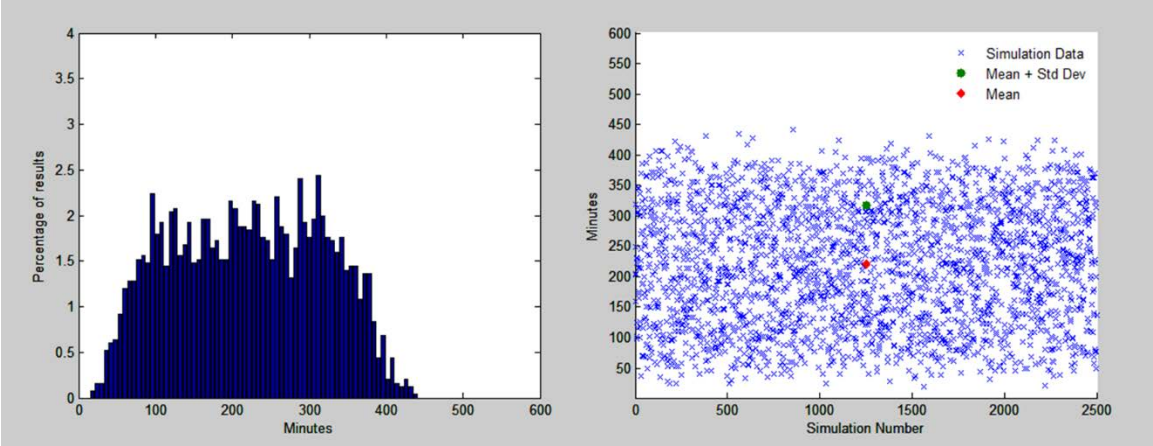


Figure 40: Results for Bed Availability Following Process Improvement

The process COA with all five control methods was evaluated and compared to the original process. The COA simulation predicted a 23% reduction in mean completion time, as well as a 22% reduction to standard deviation. However, these reductions were less than half of the improvement desired by the process improvement objective.

Compared to the recorded 12% reduction achieved, the simulated result predicted nearly double the value that was actually improved. However, uninformed guesses at the impacts of the improvement steps did manage to have results within 15% of the measured result. M&S also managed to identify that there would be improvement, but goals would not be met. In fact, simulation still showed over 45% likelihood that the timeline will still go beyond the maximum allowable turn time. Without being able to replicate the stochastic curve of each step, without being able to use the correct standard deviation for one task, and having to guess how improvement would work, accuracy demonstration is acceptable.

But why should the tool have predicted that there would be improvement without meeting the objective? There were many steps taken to reduce idle time, but not enough

targeting the actual departure time of the patient, or cleaning of the room. The sum of those two steps' mean values continued to be beyond the objective. The staff that was being trained and supplied with tools to do their part faster and more reliably would never be able to solve the issues that should have been identified in analysis. Without improving patient departure time, the mean time and deviation of the full process will still be significantly large.

In the context of predicting mean time improvement, the Process Evaluation Tool has shown a reasonable accuracy, and further ability to identify flaws in assumptions and COAs. Case study replication has also identified a critical flaw; evaluation criteria may be more than one measure, and may be difficult to quantify.

Issues Encountered During Experimentation

There were eleven issues encountered while developing the Process Evaluation Tool and experimenting with the Geico insurance claims and Our Lady of Lourdes bed availability processes.

1. Poorly defined goals
2. Non-quantitate goals
3. Poorly defined scope
4. Unavailable data
5. Too many options to create COAs for each
6. Accidently removing the best option
7. One-factor evaluation being insufficient
8. Partial factorial evaluation aliasing improvement contributions

9. Feedback loops from the last element in a process
10. Multiple start points for a process
11. Multiple exit points for a process

For the insurance claims process, the pre-defined goal in the “define” step was unrealistic. A business model would likely consider a cost model rather than a time model. For example, the option of hiring a repair shop to estimate the damages before verifying that the claim is covered would inject waste at the rate of early claim denial, and would not have been an option in a cost model over a time model. In this example, time drives cost to a degree, so a “cost” model would involve time to perform each step. The lack of a realistic objective does not impact the validity of the experiment, but does indicate that the factors in “define” need to be considered.

This corresponds to the Our Lady of Lourdes case study, where it should be assumed that there was more in the decision process than the time the process took. The data had been collected to show that the patients were the majority of the problem, but the improvement did not take the obvious step of evicting the patient. The implication is that there are potential non-quantitative evaluation criteria that may be difficult to evaluation during simulation. There was also potential to redefine the problem as the time from when the patient left the room to when the bed was available. Either of these options should be set during the “Define” step of DMAIC.

When modeling the insurance process, no data was available, and assumed data was used. While this is trivial to the experiment and was bypassed by fabricating data for the “measure” step of DMAIC. The assumptions that one insurance liability examiner handles a case and that damage estimate timeline drove the “analysis.” Following the

need to “define” the objective correctly, measuring the direct and indirect contributors to the objective in each of the tasks and relationships is recommended.

The Geico example also identified the potential for a massive optimization problem. Without even considering the ability to add or remove tasks from the process model, several billion options to analyze were present. A series of choices to eliminate options from analysis greatly reduced the amount of simulation required to search for the best option. However, the choices to eliminate options were not validated, and results are only verified on the remaining options. The hidden implication here is that each time that an option is removed from analysis, the “best” option may be removed without any indication it is no longer a solution.

Opposing simulating too many COAs, evaluating only one improvement factor at a time may miss the ability to achieve an optima that includes two or more factors. In Design of Experiments (DOE), the minimal number of tests to analyze single-factor interactions only means any two-factor interactions can not be known. Further, if something between a full factorial and a minimal factorial is run, the performance will be aliased. In other words, if each combination of potential options is not simulated, it becomes unclear which change/s provide the desired evaluation. (Hutto, 2014, pp. 4-17)

The overlap of “too many options to simulate” and simulating too few COAs becomes a risk assessment with no definitive answer. Recall that the behavior of interactions may not be able to be found using DOE, and the full factorial of all available options is the only way to ensure that the optima among factors is found (Biegler, 2010).

When verifying functionality of the Process Evaluation Tool, it is possible that a process can be modeled in such a way that it does not correctly start or stop. The simple

solution is to have a single task with no work to perform or stochastic relationships initiate the process and conclude the process, like what was added in Figure 38.

If multiple options to start or exit a process exist, the process performance may be significantly different for each start/exit combination. Change in a process model may impact each combination's evaluation differently. Attempting to make decision based on results for all combinations complicates evaluation. Evaluation of the "average" of all of the combinations confounds the potential improvement across each exit's performance. For better analysis, each combination should be binned separately prior to evaluation to isolate improvement results.

Real World Application: Identifying Incorrect Assumptions During "Control"

The final evaluation of the Process Evaluation Tool covered in this research will be application against real world issues that require process improvement. The process that was used was a combat software programming process that had already been improved, but was falling significantly short of improvement objectives.

Upon the first completion of a newly established combat software programming process, a rapid process improvement event was initiated. One of the issues targeted for improvement was an effort to expedite identification and correction of software errors. At the end of a three year process, a small analysis team would be tasked non-stop for the last eleven months, and was on the critical path for software delivery during the last eight.

The flight that owned the analysis team determined that tools to improve analysis would not be possible due to the dynamic nature of the "bugs." More analysis team

members would be necessary to reduce analysis's impact to the Squadron critical path timeline. However, adding members to the team was not easily done. Hiring additional personnel was not an option due to needing extensive experience in combat software to troubleshoot anomalies in the software. Moving individuals from one team to other teams had previously cause bottlenecks to shift without reducing the overall timeline to complete the full process.

The solution enacted was to take all members of the Flight (a United States Air Force unit below "Squadron" level) and train them to perform work under two separate teams. The plan was to have each member of the Flight have one specialized team responsibility, and also be able to program the software. The team that had been responsible solely to do the programming would be reduced to two individuals, and all members would be sent to one of the other teams in the Flight; data management, converting programed data to encrypted combat software, a field support team, or error analysis.

In theory, whenever a person was not required to do their new team's main task, they would be available to program. Because the programming team had previously been sixteen people, if seventy percent of the Flight's twenty three personnel were programming on average following the manpower change, programming would continue to complete at the same rate. At the same time, the individuals would reduce Squadron process execution time when the three other teams were on the critical path.

In the expedited timeline of a rapid improvement event, there was almost no analysis performed on this COA before implementation. Within three months, the

programming rate of the Flight was evaluated to be one third of what it had previously been. Individuals had time to program, but work was not completing efficiently.

With the improvement already performed, the process was in the "Control" step of DMAIC. It was being monitored by Flight and Squadron leadership. When the programming speed declined, a series of assumed root causes circulated. The project manager requested simulation help analyze to what was causing the significant change to production rates, and verify if the theories for reduced performance were valid.

The first criticism was that the restructure did not account for leave, sickness, training, etc. Another criticism was that people who supported two tasks would need more training and have less time to perform their main team's work and even less time to program. Additionally, if individuals were left to decide their own task priority for the day, it was assumed that they would be less likely to work on the secondary task of programming. Finally, it was theorized that by spending less time programming, programmers would become less proficient and therefore take longer to complete their work.

Building the simulation of the process was fairly simple; data had been collected for the rapid improvement event, and the updated process was documented. The Process Evaluation Tool developed in Chapter III was used with to analyze the application of manpower to the completion of tasks while executing the process. The simulator itself was modified to apply resources (in this case manpower) to priority tasks, then apply remaining manpower to lower priority tasks after all high priority work was assigned.

The previously completed Define, Analysis and measure steps narrowed the scope of analysis to simulating the improved process with a modified assumption. No

alterations to the tasks or relationships were required, and prioritization was set by the Flight. By testing each COA, assumptions that reflect the actual performance are theorized to be a potential root cause, and results that do not reflect the actual performance were theorized to be speculation. The valid assumptions would be targets for further investigation and data mining.

Simulation of the modified process required a minimal update to the developed software. To evaluate how manpower was allocated on a day-to-day basis changed the way that the tool assigned manpower to a task and completed a task, but kept the same tracking and work queueing algorithms.

To update the process to the new manpower rules, the five types of manpower remained. All but two programmers were moved to one of the four other teams (in accordance with the Flight's reorganization). To simulate the ability to flex between teams, every person on another team was also added to the programming team. When a person was working a non-programming task, manpower was deducted from the respective team and the programming team. Each task requiring programming was moved to the end of the prioritization list, so that manpower would be pulled from specialized teams. Finally, at the end of each time period in the simulation (days in this case), the list of tasks to be worked on the following day would be reset to priority order.

Simulating both the old and new manpower rules under baseline assumptions, the data showed very little to say how well the teams were being utilized, how busy they were, etc. Figure 41 and Figure 42 provide nearly no information other than dual-tasking the Flight members provided roughly a 20% decrease to the end date. Reduction in time would imply that programming may be faster.

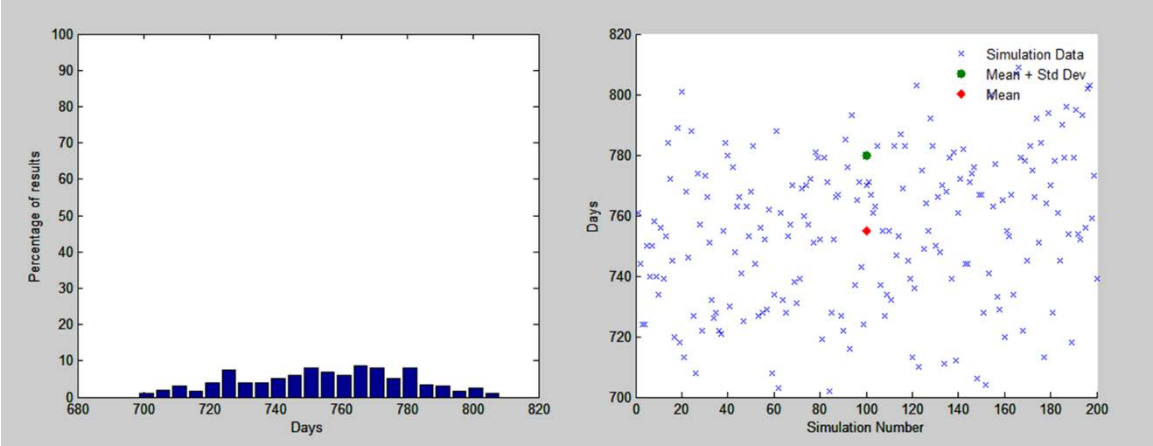


Figure 41: Legacy Process Performance with No Assumptions

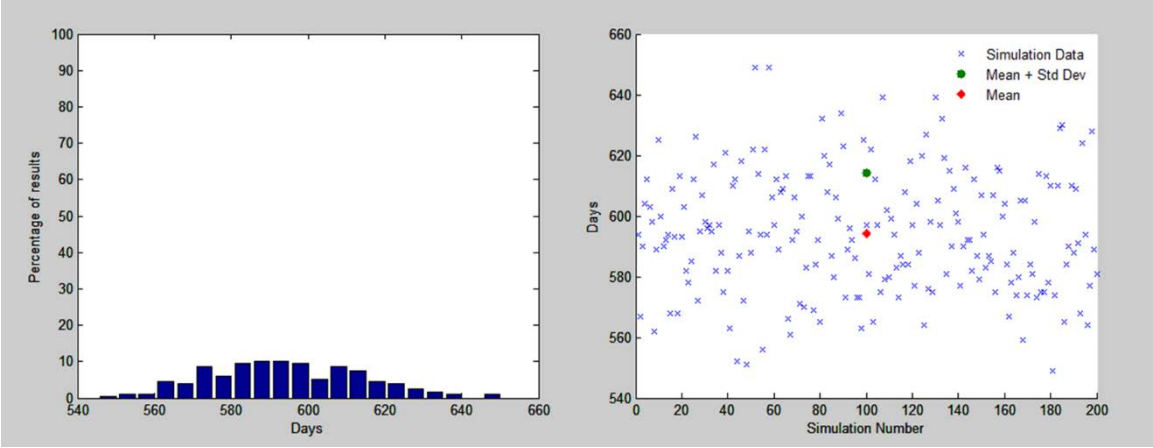


Figure 42: Process Performance after Improvement Event with No Assumptions

To get better data on the actual usage of manpower, the work queue that tracks what resources were used to perform tasks at certain times was used to view data. By looking at the variables that tracked dynamic allocation of manpower to tasks, the amount of hours worked by each team, the number of man-hours available during the entire process, and a graph of the usage for each team could be generated. Calculations against the measured data and the number of team members available were used to calculate the

amount of time that the team was fully used and holding up the critical path of the process.

In Figures 43 through 47, the Programming Team is the blue ribbon, Data Analysis orange, and the other teams largely inconsequential. The height is how many individuals had programming as a priority on any given day. By assigning each Flight member two types of tasking, it can be seen that programming is much more reactive to other efforts, but on average much higher (and therefore completing work faster) when flexible. Without assumptions, prioritizing Flight members across two tasks is a reasonable solution. As performance is not represented by this model, it is concluded that the assumptions are wrong, and theories on invalid assumptions would need to be investigated.

The first theory was that manpower was not always available. The assumption reduced the average availability of any team to 80%, plus or minus 5%. Manpower usage during simulation did not significantly change. Tasks across the board took approximately 15% longer, and the amount of task saturation experienced by each team increased only by a minor amount. The simulation was regarded as a “shoe that did not fit.”

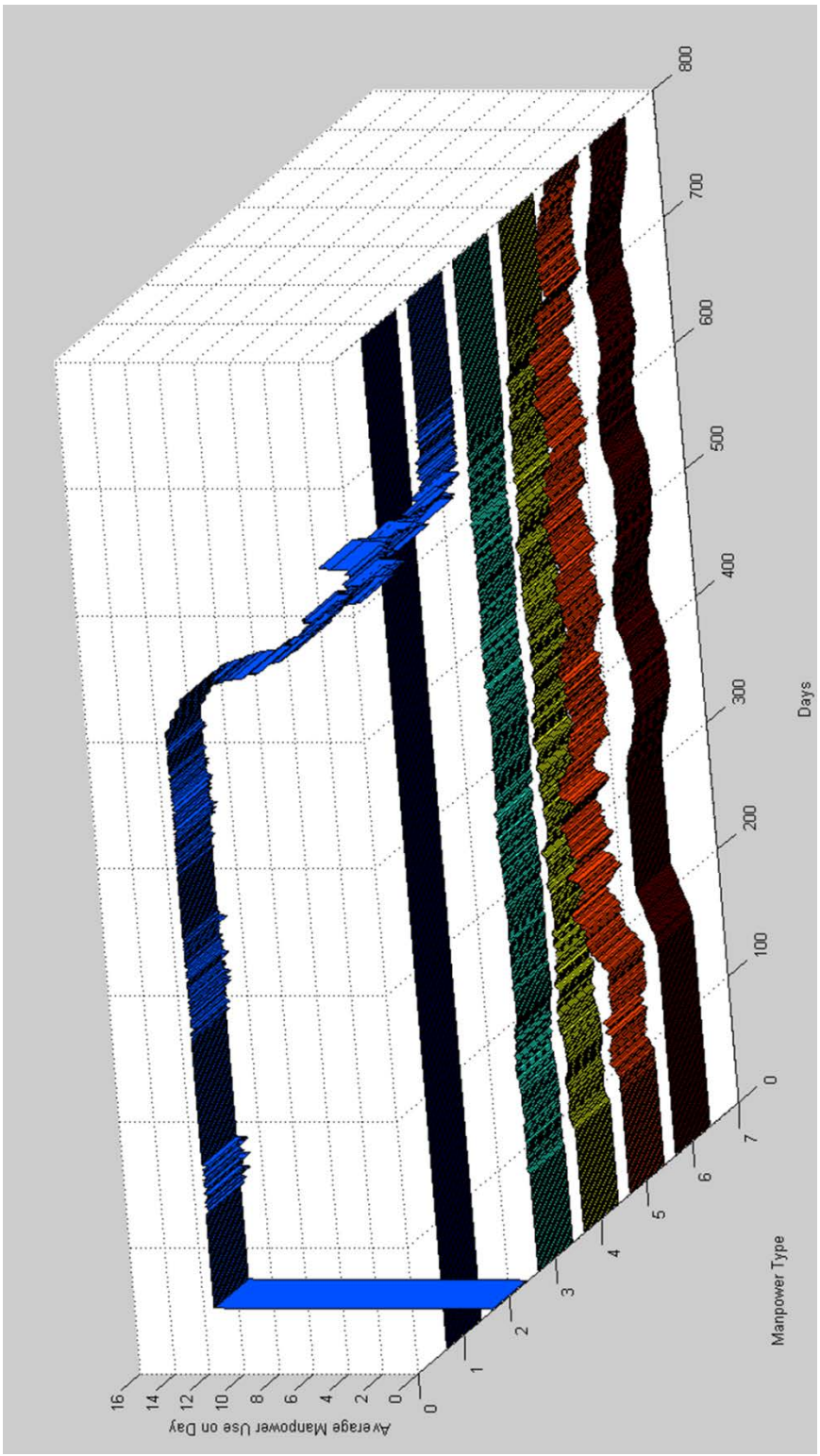


Figure 43: Legacy Manpower Usage with No Assumptions

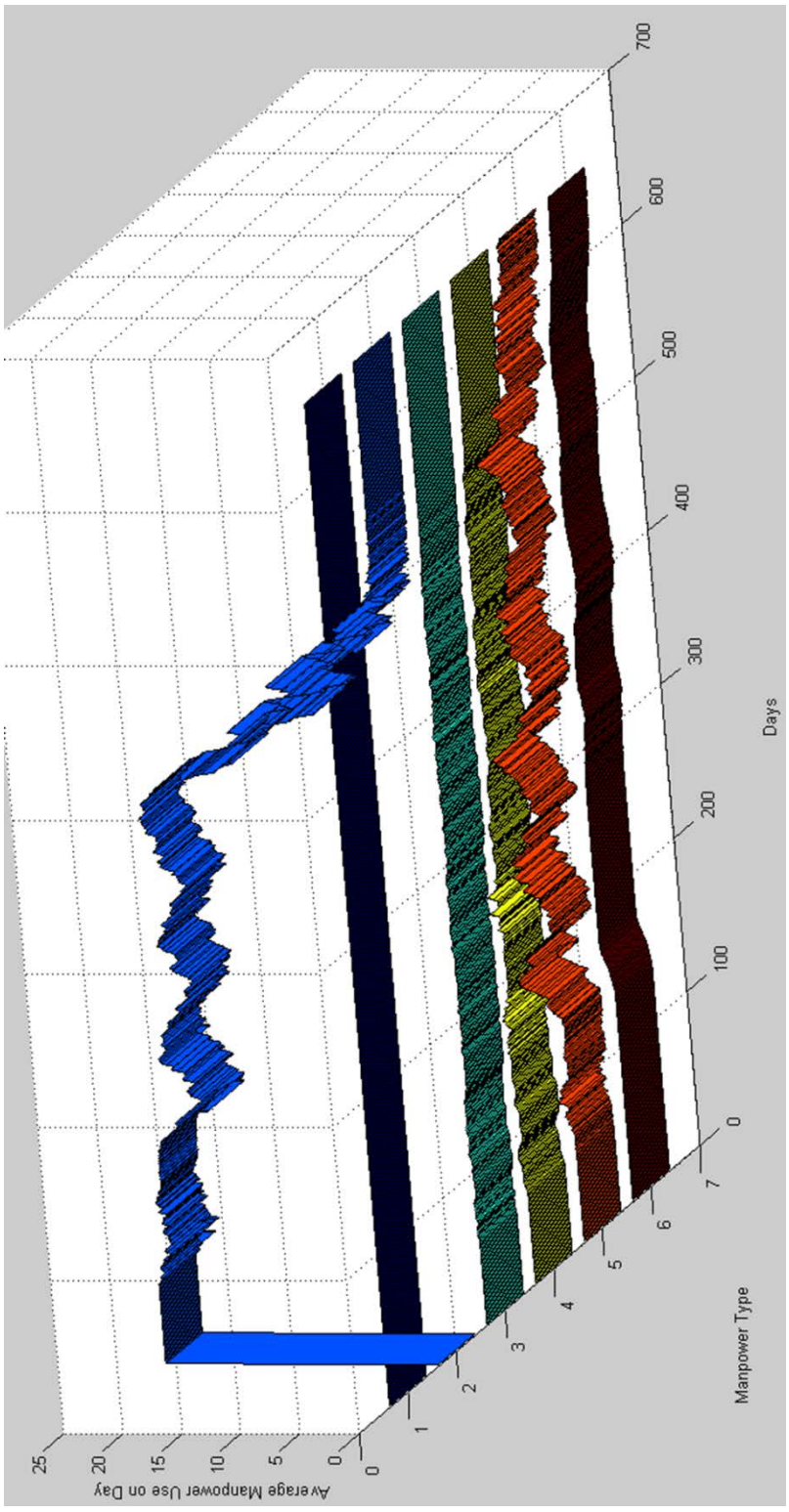


Figure 44: Updated Manpower Allocation with No Assumptions

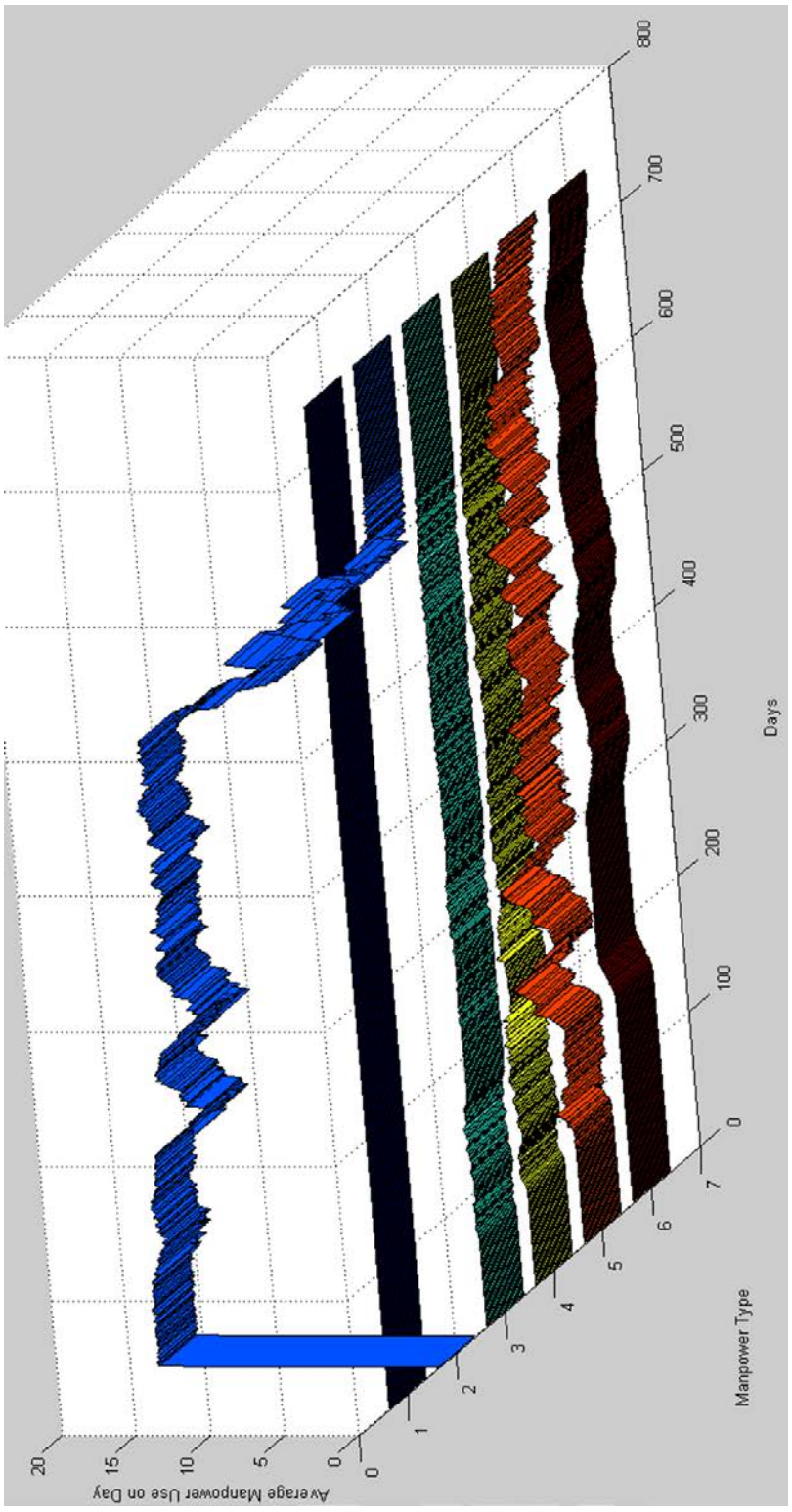


Figure 45: Updated Manpower Usage with 80% Manpower Availability

The next theory simulated was a much higher amount of unavailability due to team members need to stay up to date on two jobs' worth of training. Based on the theory, availability was dropped to a random number between 50% and 70% on any given day. Again, the timeline to complete the process increased for both cases, but the manpower usage graphs did not fundamentally change to show programming as a significant bottleneck. In this COA, the timeline increase was closer to 45%.

Both theories that simulated decreased manpower availability showed that the overall process would slow on every line of manpower. Actual process performance was moving at a regular pace on all tasks except for programming. Simulation did not show programming falling behind other steps of work to create a significant bottleneck. These assumptions did not seem to reflect the issue that was being experienced.

The third theory was that in the updated manpower allocation, workers would be less likely to program, even if their main tasks were not executing. This assumption was simulated by modifying simulation code to reduce the available manpower to program only after determining they were not executing their primary tasks on a day.

The effects of this assumption can be seen in Figure 46. Significantly less manpower would program on a given day, driving the schedule out longer. On average, the programming team was approximately six workers deep, right about one third of the assumed sixteen average. This coincided with what was being experienced, and was handed off to the remaining programming team to evaluate further.

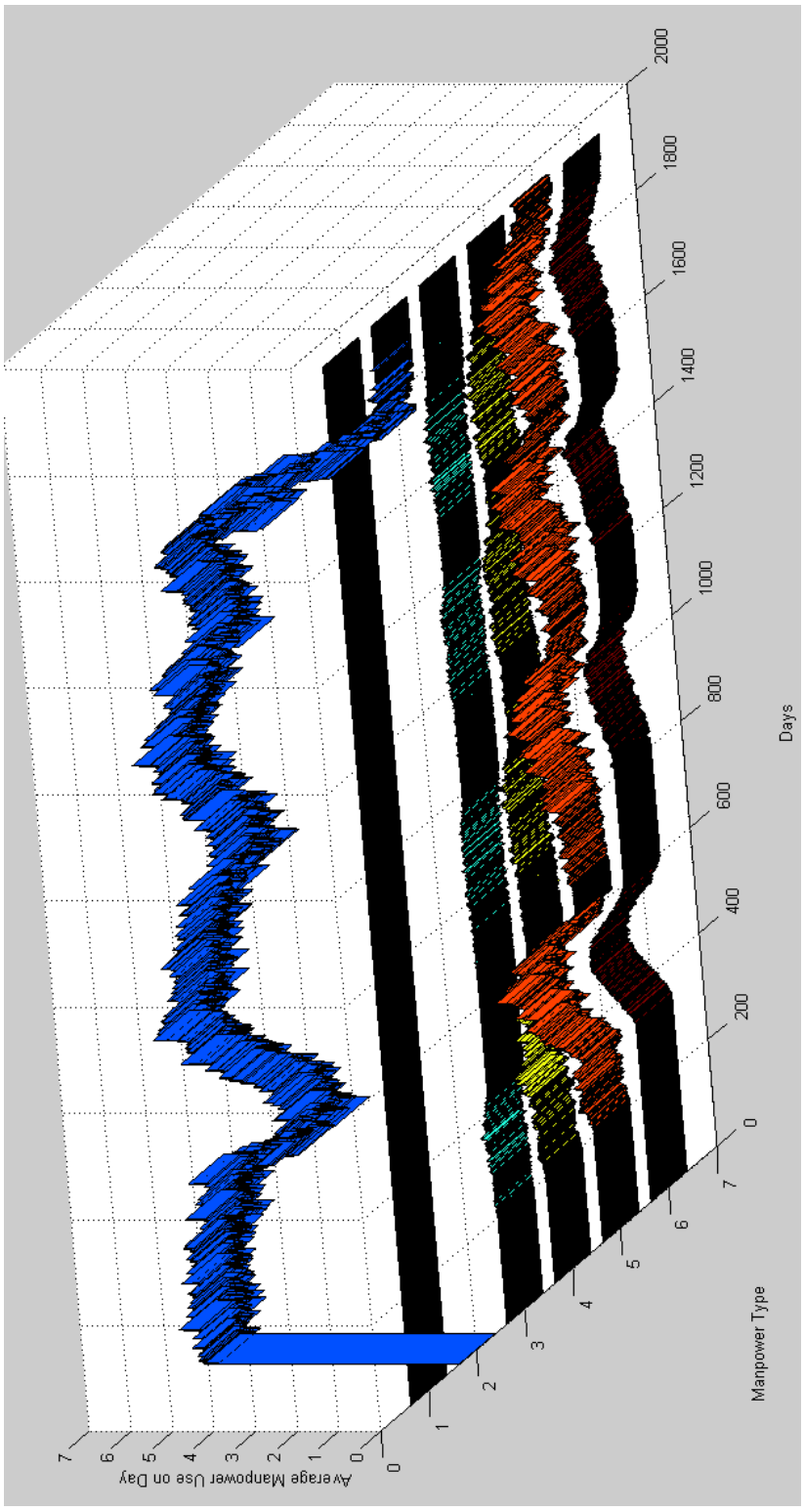


Figure 46: Updated Manpower Assuming Less Programming Prioritization

A cursory review of the flight personnel showed more than simulated participation on programming. In fact, it appeared that most available people were actively contributing to the programming effort. Conversations with personnel also invalidated the self-prioritization assumption by showing a contradiction to the simulation results; individual were busy more often than idle. These results helped invalidate the current theory and confirm that the next assumption was likely.

The final theory was that rather than reducing the number of people willing to program, the efficiency of individuals while programming would decrease. The simulation rules were updated to make manpower 100% effective on their priority tasks, but only complete part of a day's worth of work when spending a full day programming. Reducing the effectiveness of programming decreased to half, so that it took the average programmer twice as long to complete any given programming assignment. The simulation results (Figure 47) were consistent with the issues the programming team was facing, and again it was given to the team lead for in depth analysis.

Given a reasonable theory to look into, the team lead searched data to find out why efficiency could be lower. Upon review, the team lead noticed a pattern that the newer—and therefore less proficient—individuals were the one who were often freed up from specific tasking to program. The knowledgeable individuals who could quickly program complex systems were rarely freeing themselves up to program. Effectively, the manpower was less efficient in programming because of the individuals who were executing non-programming tasks.

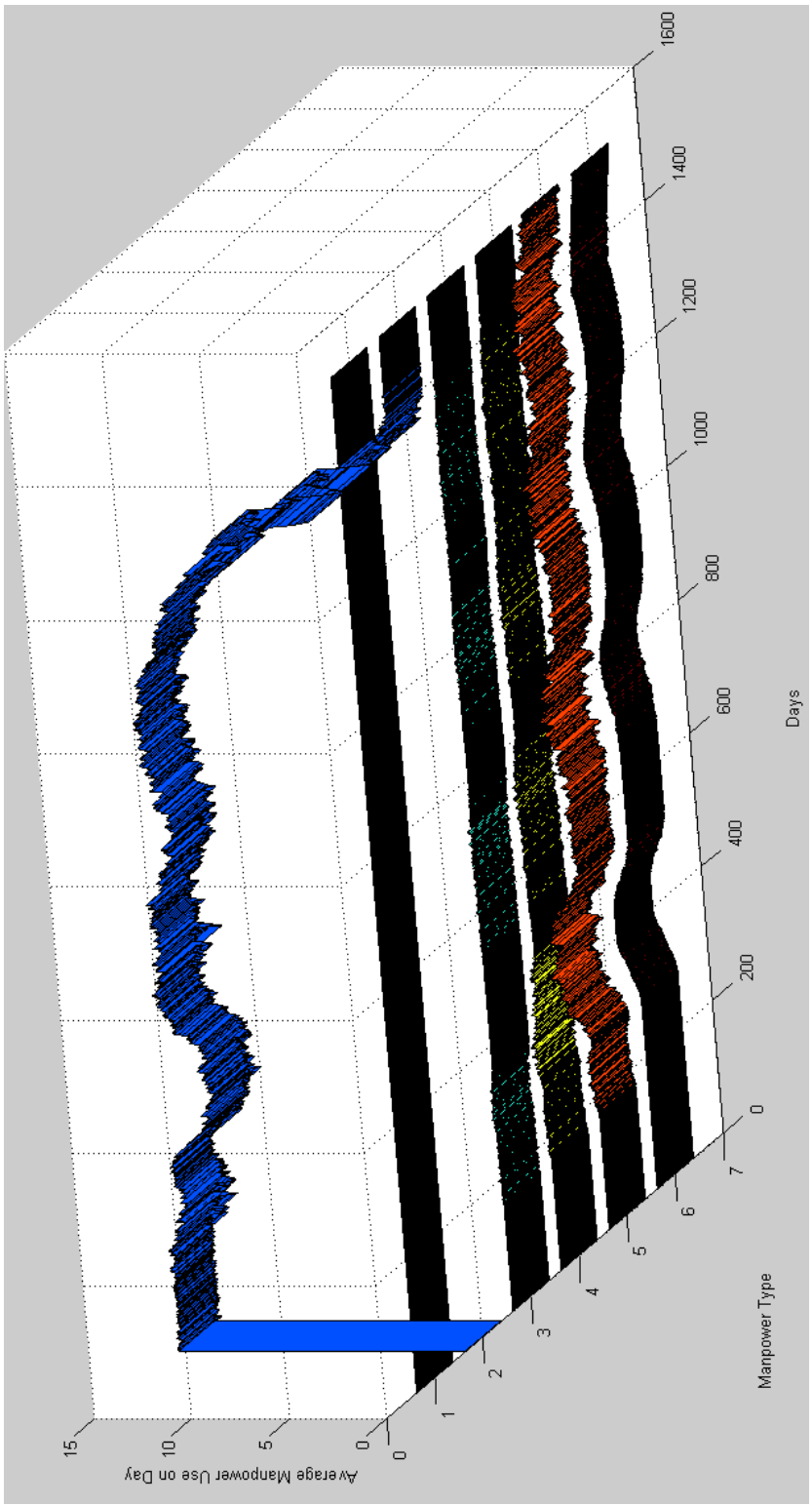


Figure 47: Updated Manpower Assuming 50% Efficient Programming

In this case study, there was a problem viewed by leadership that had four routes of speculation. The time to assess all four theories would have added approximately three weeks of delays to the work in progress. Simulation was able to establish a model, simulate each theory as a COA, and validate two theories within three days, eliminating two theories and saving an estimated seven days of delays. M&S directed the team lead's focus to identify the root cause before leadership had made recommendations on how to change the process, and before the workforce slid back into the prior way of doing business.

The root cause was brought to leadership, and the flight's team leads resolved to spread work across junior and senior team members whenever possible. Additionally, it indicated that a moderately sized team of dedicated programming personnel were still need to prevent programming from becoming a bottleneck, and began the hiring process for four personnel that operate under a different team allocation than the rest of the flight. In the DMAIC construct, the Process Evaluation Tool was able to assist in the "Control" step by applying variation to the assumptions in the process's execution. Replication of actual conditions was able to rapidly invalidate guesses at what could be causing issues, and save lengthy investigation.

Summary

Applying Modeling and Simulation to DMAIC process improvement has demonstrated the ability to evaluate COAs for expected performance with acceptable accuracy. The Process Evaluation Tool has been use to simulate a process with changes to resources, tasks, relationships, priority, and assumptions on how work is handled. The

tool has demonstrated a reasonable amount of accuracy for a process with speculative improvement expectations. It also has shown value after an improvement event's breakthrough stage, where the model found flaws in the underlying assumptions.

V. Conclusions and Recommendations

Introduction of Research

There are a variety of tools and methodologies that can greatly increase the efficiency of processes. As a result, there are many ways to change a process with the intent of improving its performance. Without knowing the resulting impact of each change, it is possible to select an improvement that is not the best option or is potentially detrimental to process improvement. To compare various option's, evaluation of each option using the same criteria can assist the process improvement decision.

In the Department of Defense, CPI/LSS applying DMAIC is a current standard for process improvement. DMAIC still allows for different means of Measurement and Analysis, and ineffective process improvement decisions can be implemented. Adding a framework to evaluate each option using Modeling and Simulation can help reduce the likelihood that bad decisions are made.

A Process Evaluation Tool was designed to assist DMAIC based process improvement. The tool implemented NDSM framework with a Monte Carlo process simulator to evaluate multiple COAs to improve a process and compare the results for various types of performance. Applying M&S bypasses several constraints with evaluation, but will only find the best COA provided to the tool.

The process evaluation tool was verified for deterministic and stochastic process behaviors, used to experiment on established process, replicated case studies, and applied to real-world process improvement events. The objective of the research was to find how

M&S can augment DMAIC to evaluate COAs, assist in finding optimal COAs, and aggregate best practices in process improvement.

Summary of Research Gap, Research Questions and Answers

The gap in process improvement is verifying that changes to a complex process will improve performance. When deciding between multiple options, determining the best option requires comparable quantification between each option. An evaluation criteria is desired that can apply to each process improvement COA, display expected results in the same manner, and apply to most methods of process improvement.

To establish a modified DMAIC methodology, a modeling and simulation tool that would evaluate process performance was created in Matlab. Experimentation was performance to answer three primary questions.

Answer to "What is the formulation to evaluate process improvement courses of action?"

The critical components to evaluate a process decision are a process improvement methodology, improvement goals, measured process data, a process model, a simulator, evaluation criteria, and one or more proposed changes to the baseline model or data. It is recommended that the method applied is DMAIC to establish the improvement goals, evaluation criteria, measure data, and proposed change/s to the process through analysis.

While executing DMAIC with the intent to verify decisions, several alterations are required at specific stages. During the Define stage, the improvement goals should be quantitative and include every factor in decision making. Computerized simulation cannot "guess" at feasibility, so every invalid option needs to be excluded from the

simulation or defined as failure criteria in the simulation's evaluation. Similarly, multiple objectives need to be quantified in relation to one another, aka a "weighting scheme" amongst desired performance metrics.

Measurement needs to collect (or approximate) all data relevant to establish a valid model of executing the process, and the data required for the evaluation criteria. These requirements are in contrast to the current implementation of the process. Relationships between steps in the process must be based on materiel or information handoffs, not "what we do next." Data must be attributed to the correct task or relationship. All relevant metrics that are used in evaluation criteria calculation need to be measured. In addition, simulation of the measured data should give representative answers to measured data--current process performance should be part of the "reasonable" simulated data.

Analysis can apply simulation once, regularly, or recursively. It is recommended to apply simulation to help identify and develop options for improving the process, and again when evaluating options for a local optima at a minimum. For example, simulation of the process can identify bottlenecks in the process's design when there are significant idle resources, allowing targeted improvement of the limiting factors.

Finally, during Control, a model has already been established with the implemented changes. Performance of the new process should have results within the simulated space. If performance is outside expectations, the model can be used to determine what assumptions in the changes were invalid or find where measured data has changed. The model can also be used to verify if additional modifications to the tasks are anticipated to be beneficial.

In addition to direct application of M&S to augment DMAIC steps directly, reviewing data while analyzing evaluation results identified an underlying pattern. If a COA evaluated favorably, there were times where a favorable evaluation was counterintuitive. Typically, these where due to a "bug" in the code that was being exploited by a COA. When reviewing favorable COAs for validity, two measures were often checked; did the data that created poor performance improve, and—in spite of the evaluation criteria—was the defined goal met?

Validity checks were being used to ensure that evaluation was favorable due to improving modeled data after applying changes to a process, and that the evaluation was meeting—if not working towards—the defined goal. When reviewing the pattern, it was seen that a strong correlation between the goal, the measured bottlenecks and root causes, and favorable evaluation indicated a strong likelihood of improvement. The alignment between Define, Measure, and Analyze indicates that the COA's changes to a process are consistent with goals and measured data.

Consistency of process improvement measures was a repeated lesson across tool development. In the context of Six Sigma process improvement, consistency is the expectation that evaluation will be used to choose the most beneficial course of action to the evaluation formula, and therefore efforts in earlier stages of process improvement should support evaluation. Applying a M&S focus across the full DMAIC method, and a robust process simulation tool establishes a reliable way to quantify the expected performance of COAs.

Answer to "How does evaluation find the best way to improve a process?"

Using simulation to verify coherency allows large quantities of COAs to be evaluated. Comparisons between multiple COAs will identify that a given set of process improvement measures provides the best expected performance, and coherency checks verify that improvement are based on logical analysis from measured data.

Optimization of a process would be ideal, but requires every potential process to be turned into a COA and passed to the evaluation tool. Rather than attempt to optimize, evaluation can compare COAs by determine a quantified value for each, and select the minimum or maximum. By bounding the problem during process improvement Definition, developing a large number of COAs, ensuring validity and coherency are taken into account during evaluation, and making a full factorial of all options, a robust set of COAs can be evaluated.

Evaluation has several benefits that assist in selecting a COA without the need to perform mathematical optimization. First, simulation allows an understanding of complex systems without needing to find objective function and solve for the minima/maxima. In addition, random sampling performed a significant number of times assists analysis. Evaluation requires a model to be established once, and can rapidly assess COAs meaning that more options can be considered than traditional calculation. Finally, by having an evaluation function, multiple criteria can be quantified, weighted, and summed, allowing comparison between COAs with dissimilar benefits against a well-defined evaluation criteria.

While not a desired answer, the evaluation tool as developed in this thesis is limited to only finding the best option simulated. Modeling a process will give more

accurate performance than approximation. Setting boundaries on the problem and goal, and accepting a local optima provides a simple means of identifying COAs that are likely to be the best course of process improvement. “Data mining” a simulator enables evaluation criteria more in line with decision making than individual calculations.

Answer to “What heuristics from evaluation guide process improvement?”

While using digital modeling and simulation to improve processes, there were several lessons learned. Most of the lessons were already incorporated in the response to the previous research questions, but several notes fell into the decision of what COA to choose. The decision criteria to determine which option is “best” is critical to understand before the “Measure” step of DMAIC, especially when seeking consistency.

Improving a process involves a decision to solve one or more issues. Often, the decision will usually have to make a tradeoff to enact the desired change. The evaluation criteria needs to be able to accommodate a trade space analysis, or computerized searches for the best improvement will not be aligned with the decision criteria. Essentially, the evaluation function should be a quantitative model that approximates the decision maker’s judgement criteria. It is recommended that the individuals components in the formula (ex. cost, schedule, performance, low variance, low defect rate, feasibility, validity, etc) each be tracked in a simulation separately, and combined into a weighted formula. After simulation the components can be reviewed so that the algorithmic “best” options can be easily seen by the weighted formula.

Consider statistical reliability in making evaluation criteria. While the output of the evaluation formula will identify a “best” option, there are cases where business case says a less optimal is better. For example, if an item is available for $92 and will deliver

within four weeks, or could be purchased with rush shipping for $99 and will arrive in five business days, are the additional three weeks of uncertainty worth $7? If the "value" of certainty can be quantified, add it to the evaluation criteria. If not, it will be a factor in decision making that was not able to be modeled, and a greater number of COAs should be considered and reviewed for the unquantifiable factor.

Model the business's view of the evaluation criteria. Time to complete a task is not measured in days or minutes, but a quantitative amount of work that needs to be performed by one or more employees who can perform it at a given rate. Rework often does not cause a return to an earlier point in the process, but launches a few tasks to be partially performed again.

Each of these recommended changes require specific "Measure" criteria. To evaluate through M&S, the data must be collected. Before measurement begins, the entirety of the decision model needs to be understood to ensure the data is collected. Consistency implies that decision criteria should be set in Define, or it may not be measured, and therefore not available for Analysis.

Recommendations for Action

The first recommendation would be to update DMAIC guidance to highlight the importance of consistency in the first three stages. Primarily, the decision criteria for evaluation be understood in Define to ensure relevant data is collected in Measure. Stating an improvement objective does not capture nuances in value-based decision making.

The decision making model includes a trade space analysis of improvement at a some cost of effort, time, capital, satisfaction, etc. The majority of the factors that weigh in on the decision making should be known before measurement begins, the relative weight of each factor should be known. By establishing the evaluation up front and planning for consistency, DMAIC improvement efforts will be concentrated towards supporting the process manager's decision.

The second recommendation is that applying M&S to process improvement should be recommended in the DoD's next process improvement guidance update. It should outline consistency through M&S, and the need to apply M&S mentality in all stages of DMAIC. It should cover benefits in large COA analysis, ability to evaluate consistently during decision making, and the ability to continually assess a process's performance in an evolving resource and constraint environment.

Recommendations for Future Research

During experimentation, two issues continued to be problematic. Different processes' execution rules would be different based on the business model they worked under, and would require recoding to simulate correctly. The second issue was that different evaluation formulas relied on different types of data, and evaluation would strongly weigh towards some COAs based on quality of data with respect to the evaluation criteria.

To avoid custom-coding a simulator for any given business model or process execution rule set, it is recommended that a research study aggregate different types of processes and generate code that handles each. For example, a well-established

production line that continuously generates products works differently than a sprint programming team, which operates differently than a delivery service. Each are valid, but apply resources and queue work differently.

A more robust simulator incorporating the ability to select different ways processes execute should be developed. By doing so, business strategy can be an independent variable when establishing COA factors. For example, a COA may be to change resourcing replicate a production line, and a predefined code for different business models could evaluate each. When developing a more robust simulation tool, it would be beneficial to add automatic bottleneck identification, and known execution rules recommendations to resolve them.

The second recommendation is to research how types of data influence the model and choices. Depending on the evaluation criteria for the improvement event, some types of data would significantly impact performance when implemented. The same type of data would at times not impact other evaluations. The evaluation results had strong reactions to data types and fidelity, based on the evaluation criteria.

Generating a correlation of what information impacts specific types of evaluation criteria will assist process improvement to be consistent. Having a set of “minimum measurement criteria and fidelity” for a given problem would help process improvement by ensuring that critical data is not excluded from the model. For process improvement events that did not establish a model of decision criteria during Define, the objective set would imply specific, relevant data should be measured. The information would be useful in preventing unmeasured performance factors from circumventing successful process improvement.

Significance of Research

M&S can improve DMAIC by providing a quantitative prediction of process performance. M&S can rapidly evaluate complex processes that would be difficult to perform by calculation. One model can evaluate several COAs under similar methodology for “apples to apples” comparison. Established models can be used continuously in Continuous Process Improvement to update model assumptions and assess future impacts in a changing environment.

The likelihood to make beneficial process improvement decisions can be improved by ensuring that the proposed change is consistent. Consistent process decisions select an improvement option that meets improvement goals, improves expected performance, and does so because the change to the process reduces the impact or severity of measured root causes. Evaluation of process performance through M&S allows a review of the simulation’s execution to locate root causes. Consistent improvements will show reduction in root cause during execution as well as improved evaluation.

In addition to improving the ability to locate and choose process improvement option that are likely to succeed, experimentation has indicated that DMAIC methods have a two-directional relationship from Define to evaluation. To be able to make the best decision on how to improve a process using measured data, the data must be measured. To ensure that the data is collected, it should be a known requirement by the end of the Define phase. The action taken in the forward DMAIC tasks of Define, Measure, Analyze leading up to a decision impact the ability to make the right decision. Consistent process improvement defines a wide range of evaluation factors that may

influence COA selection, as well as each factors' relative importance. Knowing decision criteria early increases the likelihood quality COAs are developed, and increases the likelihood that improvement will be successful.

M&S applied to DMAIC process improvement events, with a mindset to make consistent decisions will improve the likelihood that process improvement succeeds.

Bibliography

ACQuipedia. (2017, June 5). *Continuous Process Improvement (CPI) and Lean Six Sigma (LSS)*. Retrieved from ACQuipedia: https://dap.dau.mil/acquipedia/Pages/ArticleDetails.aspx?aid=3ac9f6c4-bede-4b63-9aac-7f4ba3c37162

Biegler, L. T. (2010). *Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes.* SIAM-Society for Industrial and Applied Mathematics.

Browning, T. R. (2001). Applying the Design Structure Matrix to System. *IEEE TRANSACTIONS ON ENGINEERING MANAGEMENT, VOL. 48, NO. 3*, 292-306.

Browning, T. R., & Eppinger, S. D. (2002, Nov). Modeling Impacts of Process Architecture on Cost and Schedule Risk in Product Development. *IEEE TRANSACTIONS ON ENGINEERING MANAGEMENT*, pp. 428-439.

Carrascosa, M., Eppinger, S. D., & Whitney, D. E. (1998). Using the Design Structure Matrix to Estimate Product Development Time. *Proceedings of DETC'98* (pp. 1-10). Atlanta: ASME Design Engineering Techincal Conferences.

Choo, H. J., Hammond, J., Tommelein, I. D., Ballard, G., & Auston, S. A. (2004). DEPLAN: A TOOL FOR INTEGRATED DESIGN MANAGEMENT. *Automation in Construction*, 313-326. Retrieved from Loughborough's Institutional Repository.

COSIMA. (2015). *Process Optimization Methods.* Retrieved from COSIMA: http://web.spi.pt/cosima/sites/all/downloads/R2_EN_COSIMA_Process_Optimization_methods.pdf

Defense Acqusition University. (2016, Nov 18). *Continous Process Improvement/Lean Six Sigma (CPI/LSS).* Retrieved from Defense Acqusition University: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0ahUKEwi5qfC5-brVAhUMzoMKHTCGBBkQFggsMAA&url=https%3A%2F%2Facc.dau.mil%2FCommunityBrowser.aspx%3Fid%3D294592%26lang%3Den-US&usg=AFQjCNECoBVnzL0yFmECFbjE1ijn56vDig

Different DSM Types. (2016, Nov 22). Retrieved from DSM Web: http://www.dsmweb.org/en/understand-dsm/technical-dsm-tutorial0/different-dsm-types.html

DMAIC. (2017, April 4). Retrieved from i Six Sigma: https://www.isixsigma.com/dictionary/dmaic/

Dodaro, G. L., & Crowley, B. P. (1997, May). *Business Process Reengineering Assessment Guide.* Retrieved from U.S. Government Accountability Office: https://govinfo.library.unt.edu/npr/library/gao/bprag.pdf

Hutto, G. (2014). Review of Fractional Factorial Designs. 4-17. Eglin AFB: Defense Acqusition University.

Larman, C. (2005). *Applying UML and Patterns.* Dehli: Pearson Education, Inc.

Learn About The Complete Insurance Claims Process. (2016, Oct 23). Retrieved from Geico: https://www.geico.com/claims/claimsprocess/

McCord, K. R. (1993, May). Managing the Integration Problem in Concurrent Engineering. Massachusetts Institute of Technology.

McGrath, E. A. (2009, July 17). *Implementation and Management of the DoD-Wide Continuous Process Improvement/Lean Six Sigma (CPI/LSS) Program.* Retrieved from Executive Services Directorate: http://www.dtic.mil/whs/directives/corres/pdf/501043p.pdf

Nave, D. (2002, March). *How to Compare Six Sigma, Lean and the Theory of Constraints.* Retrieved from Lean Enterprise Institute: https://www.lean.org/Search/Documents/242.pdf

Process. (2017, Jul 1). (Merriam Webster) Retrieved April 3, 2017, from Merriam-Webster.com: https://www.merriam-webster.com/dictionary/process

Sager, R., & Ling, E. (2017, August 1). *Leveraging Six Sigma to Improve Hospital Bed Availability*. Retrieved from iSixSigma: https://www.isixsigma.com/new-to-six-sigma/dmaic/leveraging-six-sigma-improve-hospital-bed-availability/

Steward, D. V. (1981). The Design Structure System: A Method for Managing the Design of Complex Systems. *IEEE Transactions on Engineering Management, Vol. 28*, 71-74.

Thebeau, R. E. (2001). Knowledge Management of System Interfaces and Interactions for Product Development Processes. Massachusetts Institute of Technology.

Vorne. (2016). *Theory of Constraints*. Retrieved from Lean Production: http://www.leanproduction.com/theory-of-constraints.html

Yassine, A. A., & Falkenburg, D. R. (1999). A Framework for Design Process Specifications Management. *Journal of Engineering Design*, 223-234.

Yassine, A. A., Whitney, D. E., Lavine, J., & Zambito, T. (2001, September). Do-It-Right-First-Time (DRFT) Approach To DSM Restructuring. *ASME 2000 International Design Engineering Technical Conferences* (pp. 1-8). Baltimore: Massachusetts Institute of Technology Engineering Systems Division.

Yassine, A., Jolekar, N., Braha, D., Eppinger, S., & Whitney, D. (2002). *Information Hiding in Product Development: The Design Churn Effect.* Cambridge: January.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of the collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to an penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	
16-09-2017	Master's Thesis	June 2014 – September 2017

TITLE AND SUBTITLE

Evaluating Process Improvement Courses of Action Through Modeling and Simulation

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Owens, Joseph R.., Captain, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)
Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way, Building 640
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENV-MS-17-S-049

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
513 ELECTONIC WARFARE SQUADRON
130 Wacissa Rd, Eglin AFB, FL 32542
(850) 883-0513
ATTN: Lt Col Patrick Mathis

10. SPONSOR/MONITOR'S ACRONYM(S)

513 EWS

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRUBTION STATEMENT A. APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES
This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT
Quantifying an expected improvement when considering moderate-complexity changes to a process is time consuming and has potential to overlook stochastic effects. By modeling a process as a Numerical Design Structure Matrix (NDSM), simulating the proposed changes, and evaluating performance, quantification can be rapidly accomplished to understand stochastic effects. This thesis explores a method to evaluate complex process changes within Six Sigma DMAIC process improvement to identify the most desirable outcome amongst several improvement options. A tool to perform the modeling and evaluation is developed. This process evaluation tool is verified for functionality, then is demonstrated against generic processes, a case study, and a real world Continuous Process Improvement event. The application of modeling and simulation to improve and control a process is found to be a positive return on investment under moderate complexity or continuous improvement events. The process evaluation tool is demonstrated to be accurate in prediction, scalable in complexity and fidelity, and capable of simulating a wide variety or evaluation types. Experimentation identifies the importance of understanding the evaluation criteria prior to "Measurement" in DMAIC, which increases the consistency of process improvement efforts.

15. SUBJECT TERMS
Modeling Simulation Six Sigma DMAIC Process Evaluation

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			John Colombi, AFIT/ENV
U	**U**	**U**	**U**	114	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 255-3636, x3347 (john.colombi@afit.edu)

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18